DEVELOPMENT OF A MULTIDISCIPLINARY DESIGN ANALYSIS FRAMEWORK FOR UNMANNED ELECTRIC FLYING WINGS

A Dissertation
Presented to
The Academic Faculty

By

William Valentin Whitmore

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in the
School of Aerospace Engineering

Georgia Institute of Technology
December 2019

DEVELOPMENT OF A MULTIDISCIPLINARY DESIGN ANALYSIS FRAMEWORK FOR UNMANNED ELECTRIC FLYING WINGS

Approved by:

Dr. Dimitri Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Neil Weston
School of Aerospace Engineering
Georgia Institute of Technology

Mr. Carl Johnson
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: November 21st, 2019

ACKNOWLEDGEMENTS

I would like to thank my academic adviser, Dr. Dimitri Mavris, for providing me with a graduate experience that combined exceptional teaching with practical professional development. I feel privileged to have been a part and beneficiary of the program and laboratory he directs with such unrelenting energy.

This thesis emerged out of a research project led by Dr. Neil Weston. He encouraged me to develop my work for it into a Master's thesis. For this encouragement and his steady subsequent technical guidance, I am deeply thankful.

Furthermore, I thank Mr. Carl Johnson for his invaluable assistance in the development of the electric propulsion module. I could always rely on his patience and helpfulness.

I cannot neglect mentioning my colleague Mr. Christopher Kitson, who provided many suggestions for the improvement and further development of the work I shall here present. Dr. Zhimin Liu also provided regular technical feedback, and Ms. Adrienne Durham was exceptionally helpful in guiding me around the bureaucratic aspects of obtaining a graduate degree.

In addition, I wish to acknowledge my classmates and colleagues at the Aerospace Systems Design Lab, whose companionship helped make the many challenges I faced more bearable. In that vein, I also thank my friends at Outdoor Recreation Georgia Tech, especially those in the whitewater kayaking sport group. Without my weekly trips to the river with them, I doubt I would have been able to work as steadily on this thesis as I did.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Small-scale subsonic unmanned aerial vehicles have become common tools in both military and civil applications. A vehicle configuration of special interest is the flying wing (aka all-wing or tailless aircraft). This configuration can potentially reduce drag, increase structural efficiency, and decrease detectability. When combined with an electric propulsion system, it produces no observable emissions and possesses fewer maintenance issues. Unfortunately, strong couplings between disciplinary analyses hinder the design of unmanned electric flying wings. In particular, achieving adequate stability characteristics degrades the aerodynamic efficiency of the vehicle, and constrains the available volume in which subsystem components may be placed. Exploiting the potential advantages of electric flying wings therefore necessitates a multidisciplinary perspective.

In order to overcome the identified challenges of unmanned electric flying wing design, a multidisciplinary design analysis framework was conceptualized, implemented, and evaluated. The Python-based framework synthesizes automated analysis modules that model geometry, weight distribution, electric propulsion, aerodynamics, stability, and performance. Virtual experiments demonstrated the framework's utility in quickly exploring a wide design space and assessing design robustness. Two important stand-alone contributions developed for the framework are (1) an algorithm for densely packing battery cells within a wing shape and (2) a parametric electric propulsion analysis code. In short, the framework supports the design of small-scale (i.e. 0-55lb weight range) subsonic unmanned electric flying wings with a host of valuable capabilities that were previously unavailable within traditional design methods.

CHAPTER 1. INTRODUCTION AND MOTIVATION

1.1 Unmanned Aerial Vehicles (UAVs)

Due to their countless application scenarios and low overall costs, unmanned aerial vehicles have become increasingly popular in both military and civil circles. An unmanned aerial vehicle, also known as "UAV", is an aircraft that has no humans onboard. It may be either remotely or autonomously piloted, and is part of a larger unmanned aerial system, or "UAS", that also includes a ground control station, user interface software, radio transponders, and antennae [1].

1.1.1 Advantages of UAVs

UAVs offer notable advantages over manned aircraft. First, eliminating the pilot, crew, and passengers expands the design space. For instance, the maximum load factor the vehicle may experience is no longer constrained by considerations for the physical health of any onboard humans [2]. The vehicle may also be designed for extended endurance missions that a human could not comfortably accomplish. Moreover, UAVs do not require any volume for seats, instrument consoles, bodies, and other human life support equipment. Eliminating this volume reduces weight, enhances performance, and, most importantly, enables smaller scale applications.

In addition, unmanned vehicles offer considerable cost reductions. For example, no time or resources need be allocated for the design, manufacture, testing, and certification of any components related to human life support, safety, and comfort. This drastically shortens vehicle development times, leading to additional cost savings.

Finally, UAVs reduce risk, as no human pilots or crew will be exposed to the dangers of a mission [3]. Most obviously, crashes do not place vehicle operators at risk of injury or death. Furthermore, the reduction in cost relaxes the development process's financial risk, making UAVs an ideal testing platform for new configurations and technologies.

1.1.2 Applications of UAV's

In military contexts, UAVs spare pilots from flying dull (i.e. long and uneventful), dangerous (e.g. flying in enemy territory), or dirty (e.g. flying in contaminated areas) missions [4]. These mission types are also encountered in scientific research, making UAVs excellent platforms for taking measurements in remote, perilous, or inaccessible areas. Military drones are often used in hunter-killer missions, for which vehicles like the General Atomics MQ-1 Predator or MQ-9 Reaper serve as poster children. Finally, UAVs are mainstays of intelligence, surveillance, and reconnaissance missions that provide information essential to both strategic and tactical decision-making.

Outside their obvious military utility, UAVs have significant and varied civil applications, including border patrol, fire control, construction and surveying, agriculture, law enforcement, package delivery, and filmmaking. Commercial UAS is a growing industry, benefitting from significant investment on the part of technology companies and venture capital. In 2017, funding of commercial UAS reached $511 million in the United States according to Teal Group, an aerospace market analysis company [5]. As depicted in Figure 1, the number of operating UAVs is projected to increase from roughly 3 million in 2017 to 7 million by 2026. These forecasts indicate that UAVs are quickly becoming quotidian tools in many civil work environments.

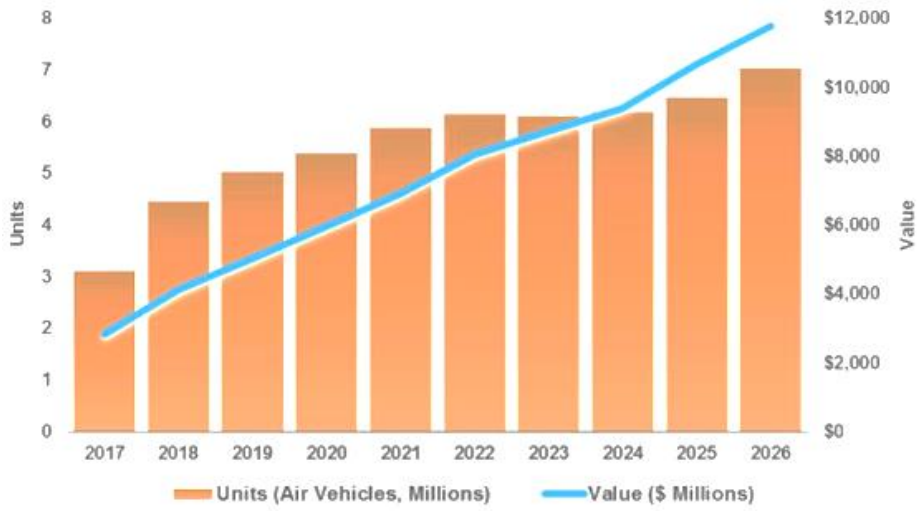

Figure 1: World civil UAS production forecast [5]

1.1.3 UAV Summary

In short, it is difficult to dispute the relevance of unmanned aerial vehicles within the field of aerospace engineering. Military reliance on UAVs is growing steadily. Moreover, the UAV industry has significant growth expectations in the coming years as civil applications of UAS expand. Within such a competitive environment, it is advantageous to develop design analysis tools that identify optimized designs in short amounts of time.

1.2 Flying Wing: Promise and Challenges

Among the many aircraft configurations, the flying wing deserves special research attention. For one thing, it is a common configuration choice for unmanned aerial systems [3]. But more crucially, it possesses attractive potentials that cannot fully be realized due to inherent design challenges, for which no standard solutions exist.

1.2.1 Definition of Flying Wings

Flying wings are also known as tailless or all-wing aircraft. A seminal work on their theory and design is that by Nickel and Wohlfahrt [6]. In their treatment, the authors define flying wings as aircraft that consist solely of a single lifting surface. This contrasts with

other configurations, such as conventional, tandem, or canard aircraft that have two surfaces located one behind the other (see Figure 2). It should be noted that flying wings may still have a vertical stabilizing surface.

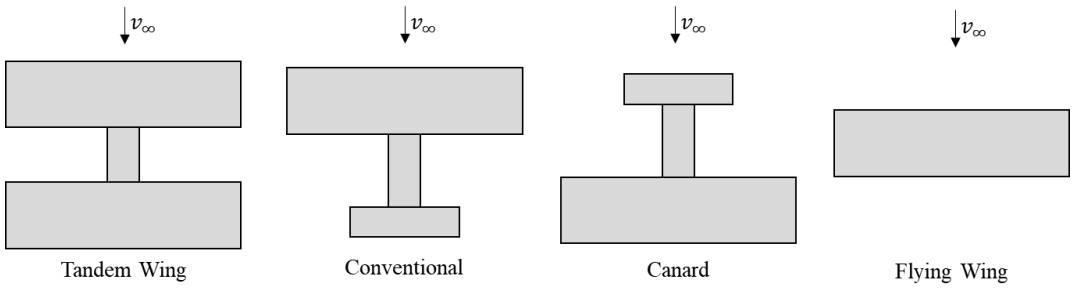

Figure 2: Common aircraft configurations

Even within the flying wing configuration, various subcategories exist. This thesis specifically addresses flying wings with blended or nonexistent fuselages, as such vehicles are of interest due to their reduced radar cross section. Moreover, stability and control issues are more difficult to resolve if a fuselage is excluded, making the design of these vehicles more challenging and hence in greater need of analysis tools. The two principal vehicle geometries examined within this thesis are the "flying plank" and the "sweptback all-wing". Figure 3 depicts these configurations.

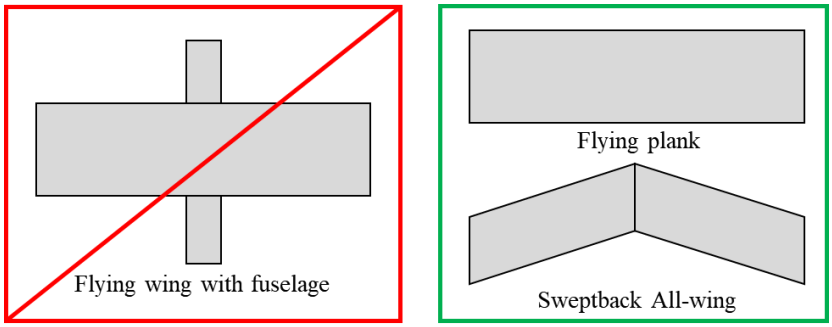

Figure 3: Flying wing subcategories considered in this thesis

1.2.2 All-Wing Advantages

Flying wings possess several inherent advantages over conventional aircraft. First, they can theoretically improve vehicle aerodynamics [7]. To illustrate, the absence of a

fuselage or horizontal or vertical stabilizer reduces wetted area. This in turn eliminates sources of parasite drag. Interference drag is similarly minimized as there are no areas where two different bodies join. As a result, flying wings offer increased aerodynamic efficiency by removing drag contributors, making them attractive configurations for missions requiring extended range or endurance.

Furthermore, the removal of tail surfaces and fuselage structure can lead to significant weight reductions. These in turn enhance performance and reduce costs [8]. In addition, flying wings can achieve the structurally efficient condition of spanloading, in which the weight distribution mirrors the lift distribution [9] (see Figure 4). Such a scenario reduces the internal loads of structural members, which can be exploited to diminish the structural weight of the aircraft. The Helios Prototype solar-powered flying plank developed for NASA's ERAST project in the early 2000s exemplifies the spanloading concept [10].

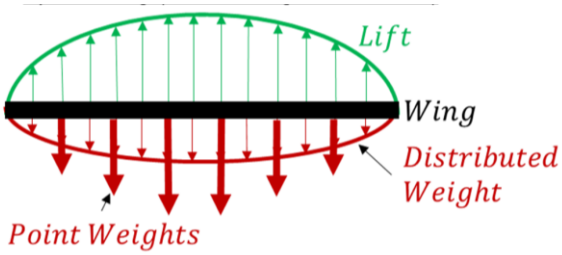

Figure 4: Graphical representation of spanloading

The elimination of substructures also simplifies manufacturing and assembly, while simultaneously making it easier to package and store the vehicle in between missions [11]. Moreover, it can reduce the vehicle's radar cross section and consequently its detectability [8]. This is a particularly important consideration in military applications, where stealth has become a vital vehicle requirement. The B-2 Spirit, for instance, fully exploits this benefit [12].

1.2.3 Longitudinal Stability Challenges

Despite the potential advantages offered by pure flying wings, comparatively few have been developed in practice. Put simply, all-wing vehicles have significant stability, handling, and control deficits that hamper their performance capabilities compared to conventional aircraft. In particular, the requirement of longitudinal stability places noteworthy constraints on the choice of wing geometry and the overall aerodynamic efficiency. Static longitudinal stability of a flying wing can only be achieved if the moment coefficient about the wing's neutral point pitches the wing up (see section A.5 in the appendix for the derivation of this requirement). There are two design choices that provide a positive moment coefficient, but they both require sacrifices in aerodynamic efficiency:

The first option is to use a reflex airfoil in conjunction with a rectangular planform. Such a shape is referred to as a "flying plank", and its conceptual derivation from a conventional aircraft can be seen in Figure 5. Reflex airfoils with their characteristic "S"-shape offer positive pitching moments, but with a loss in lifting capability and ergo aerodynamic efficiency.

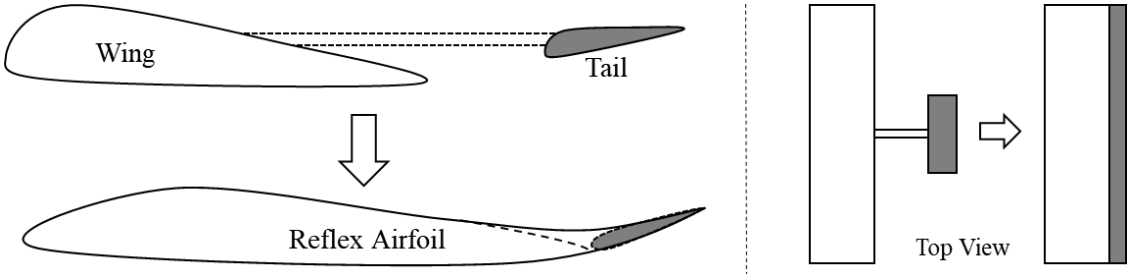

Figure 5: Evolution of the flying plank from a conventional aircraft

The second design option to ensure longitudinal stability is a sweptback wing planform with washout (see Figure 6). Washout means that there is either geometric or aerodynamic twist in the wing, such that the tips generate less lift than the root. The washout and sweep allow more lift to be generated at the root ahead of the neutral point, thereby creating a

pitch-up moment. However, the design is still aerodynamically inefficient, since the tips do not produce as much lift as they could.

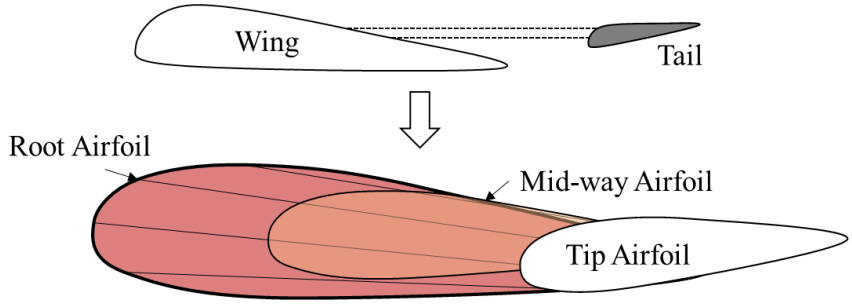

Figure 6: Evolution of the sweptback all-wing from a conventional aircraft

An additional challenge to achieving longitudinal stability arises from constraints on weight distribution. Achieving a positive static margin is more difficult on an all-wing aircraft, because there is less space ahead of the neutral point in which to place mass. Most of the subsystem components must be concentrated as far forward as possible, and the actual usable volume in the wing may be considerably less than the geometric volume [13].

Thus, there are ways to ensure longitudinal stability of flying wings, but these come at a cost to aerodynamic efficiency. An alternative solution to this challenge is to develop a good stability augmentation system for the vehicle later in design. While such a philosophy might lead to higher performance vehicles, it is also inherently risky. Ignoring stability and control concerns in conceptual and preliminary design may lead to a vehicle prototype with fatal flaws that even a sophisticated stability augmentation system has difficulty ameliorating. A wiser and less risky design approach would incorporate stability and control concerns early on in the process.

1.2.4 Lateral Control Challenges

An additional disadvantage of flying wings comes in the area of lateral control. Sweptback flying wings do have a natural, but weak yaw stability [11]. But without

conventional vertical stabilizers or rudders, the flying wing can offer only weak control power in yaw [8]. Vertical stabilizers may be added to an all-wing configuration, but they can only be located a relatively short distance behind the vehicle's center of gravity. The stabilizer would therefore need a large area to be effective, which in turn increases drag and radar cross section. This negates two of the aspired advantages of the flying wing configuration and is hence undesirable. An overview of alternative methods for providing yaw control on tailless aircraft can be found in [6] and [8]. However, none of these are as effective or efficient as employing a vertical stabilizer. To mention just one example, the B-2 generates a yaw moment by increasing the drag on one wingtip, which is clearly an inefficient control method.

Put briefly, there is no clear-cut solution to achieving good yaw control power on tailless aircraft. The advantages of the flying wing configuration are therefore best realized in applications where rapid maneuvering and strong yaw control power are of marginal importance, such as in lengthy surveillance missions.

1.2.5 Flying Wing Challenges Summary

From this review, it becomes clear that a flying wing's advantages in aerodynamic efficiency, low detectability, and structural efficiency are counterbalanced by several stability, control, and handling deficits. Longitudinal stability issues can be resolved through proper selection of airfoil, wing sweep, wing twist, and center of gravity placement, but these design choices invariably lead to reduced aerodynamic efficiency. Lateral control deficits, by contrast, are less solvable. Subsequently, lateral control issues will be treated as outside the scope of the present research, and it will be assumed that vehicle applications under examination will not require intense maneuvers.

In short, flying wings exhibit much stronger couplings between aerodynamics, weight distribution, and stability than conventional configurations. These interactions must be considered early in design if the aim is to arrive at optimized vehicles.

1.3 Electric Aircraft

The electrification of aircraft propulsion and power is a major push in contemporary aerospace research. In simple terms, electric propulsion denotes using batteries to power electric motors that drive thrust-generating propellers. The shift towards electrification is a reaction against the climate changing effects connected to fossil fuel emissions [14].

Apart from reducing environmental impacts, electric motors are advantageous because they require little maintenance and can be readily stopped and restarted during flight. Such an ability is especially useful for enhancing safety when hand-launching an aircraft. In addition, electric motors generally achieve higher power-to-weight ratios and higher efficiencies than internal combustion engines, thereby permitting weight savings and distributed propulsion concepts [15]. Furthermore, the performance of electric motors does not vary significantly with altitude, which simplifies modeling [16].

One of the main obstacles to introducing electric power on aircraft are the drastically inferior energy storage characteristics of batteries. The energy density (energy per unit volume) of lithium-ion batteries, which are currently the most advanced commercially available chemistry type, is roughly 18 times less than that of kerosene. Similarly, their specific energy (energy per unit mass) is approximately 60 times less (see Figure 7) [14].

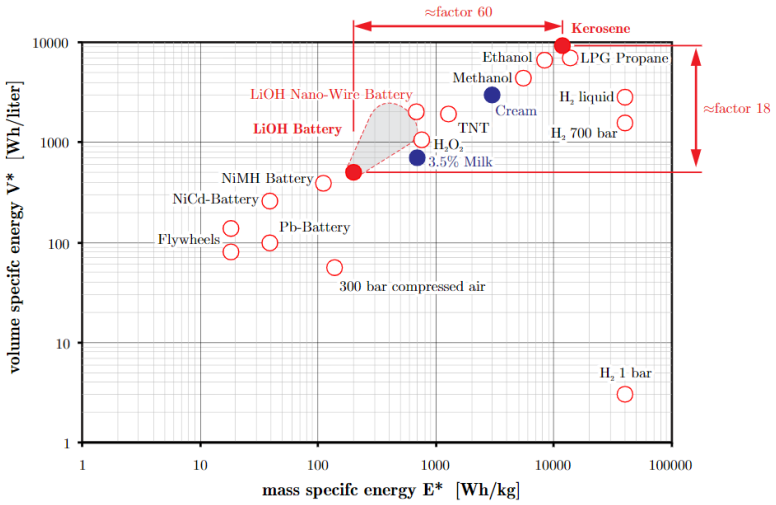

Figure 7: Energy density vs. specific energy of existing energy storage devices [14]

It is therefore evident that electric propulsion on unmanned aircraft has both benefits and challenges. The successful development of an electric propulsion system depends on the existence of good modeling tools and techniques within the design process, accompanied by physical testing. To complicate matters, the physics of electric propulsion contrast strongly with those of tried-and-true combustion-based propulsion. Consequently, understanding the system-wide impacts of electric propulsion requires a modified modeling approach:

The most obvious feature of electric propulsion is that aircraft weight does not vary as energy is consumed. But an even more consequential feature is the phenomenon of battery discharge. This phenomenon is illustrated in Figure 8, which shows how the voltage of a typical battery cell decays nonlinearly as the charge is used up, and as more current is drawn. Appropriately modeling such effects is crucial to providing accurate performance estimates.

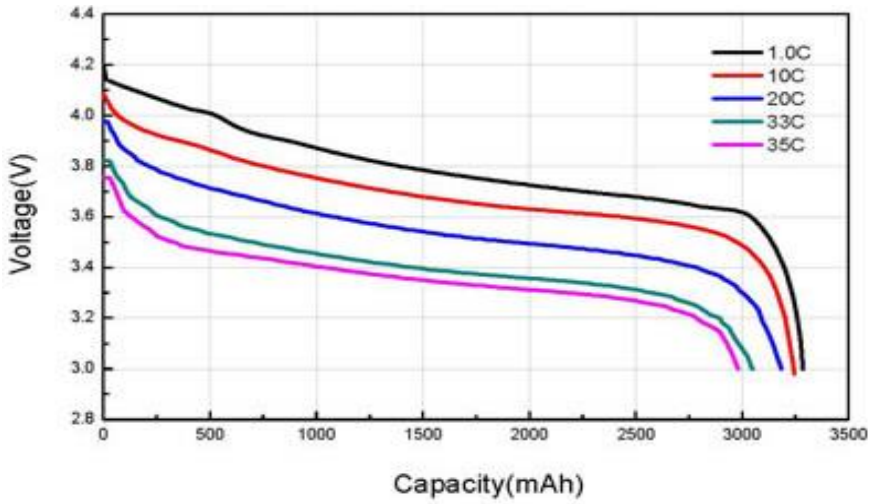

Figure 8: Typical lithium-ion battery discharge curve at different current draws [17]
(larger C-rate means higher current draw)

Moreover, power availability interacts with the geometry, stability, and aerodynamics of flying wings. This dependency results from the need to place propulsion system components, especially batteries, inside the wing in an arrangement that achieves a positive static margin. Batteries usually come in discrete rectangular prismatic size options that do not readily conform to the complex shape of a wing. It quickly becomes evident that different airfoils and planform shapes offer more or less advantageous positioning options for the batteries. The wing geometry choices in turn affect aerodynamics, which impacts power requirements, which affect the number of batteries needed, which, coming full circle, need to be arranged within the wing shape in a way that ensures some degree of stability. Consequently, introducing electric power into the flying wing configuration adds further discipline couplings.

1.4 Motivation Summary

This chapter has provided an overview of unmanned aerial vehicles, flying wings, and electric propulsion for aircraft. Unmanned electric flying wings have a variety of potential benefits and application areas. However, interdisciplinary analysis couplings between

aerodynamics, stability, weight distribution, and electric propulsion complicate their design. Therefore, the development of tools to overcome these design challenges is a relevant area of research:

General Research Objective:

Develop tools to aid in the design of unmanned electric flying wings

To narrow down the research scope, the design tools focused on small-scale subsonic aircraft in the weight range of 0-55 lb. The tools were intended for use in late conceptual and early preliminary design.

CHAPTER 2. BACKGROUND REVIEW

2.1 Existing UAV Design Methodologies

The first step towards uncovering improved design aids for unmanned electric flying wings is to define the features and shortcomings of current UAV design methodologies. Unfortunately, design practices in industry are proprietary. Nevertheless, design reports from academic competitions such as AIAA's Design Build Fly are available and offer insights into design techniques employed on small-scale electrically powered unmanned aircraft [18]. These design documents reveal some common features of UAV design methodologies:

2.1.1 Physics-Based Approach

Unlike some unconventional configurations that suffer from a lack of historical data, UAVs are characterized by an overwhelming diversity of data. Countless designs have been developed and manufactured due to their low costs. The aircraft differ in their sizes, configurations, energy sources, structural layouts, materials, and overall quality. This diversity makes it challenging to develop accurate regressions based off existing data. A historical-data-based approach to vehicle design is therefore not often appropriate, nor is it particularly desirable, as one of the exciting aspects of developing UAVs is the ability to push boundaries with little risk. Clearly, a physics-based design method is necessary.

2.1.2 Fidelity Spiral

Within a physics-based approach to design, the fidelity of analysis tools commonly increases as the design process progresses. At the outset, very simple relations or assumptions might be used to estimate vehicle aerodynamics and propulsive efficiency.

Later in preliminary design, after more of the design has been frozen, slower, higher fidelity tools, such as computational fluid dynamics or finite element analysis will enhance the knowledge of the design. This leads to a negative correlation between design freedom and design knowledge, as visualized in Figure 9. Maintaining design freedom while also gaining knowledge and arriving at a final design in a short timeframe is generally quite challenging.

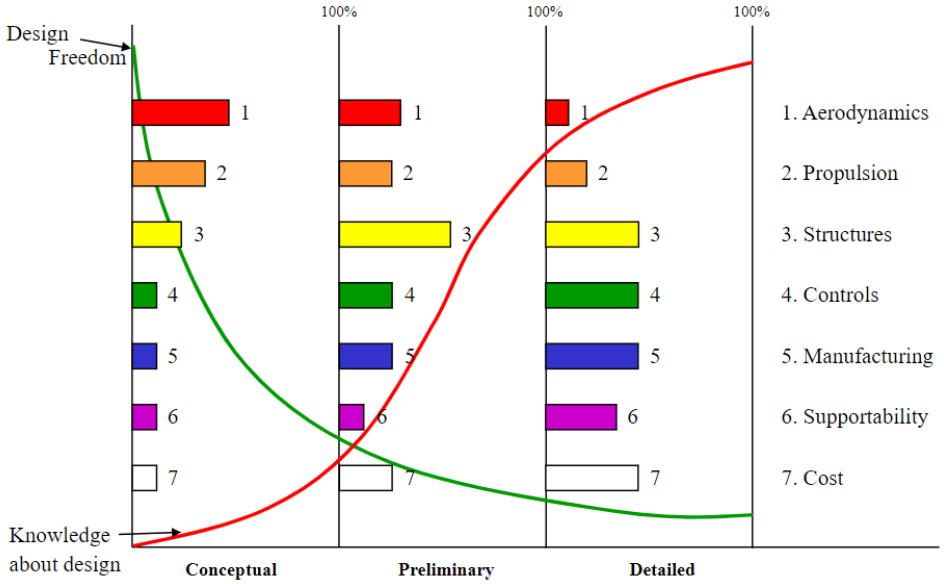

Figure 9: Notional design freedom and knowledge over the course of a traditional design process [19]

2.1.3 Separation of Disciplines

There is a tendency in design to separate aeronautical disciplines. Design teams are commonly divided into subgroups, each responsible for a different discipline, such as aerodynamics, propulsion, structures, stability and control, and manufacturing. Such a division of labor works best if the disciplines are mostly independent of one another, meaning that they do not share inputs, and that the outputs of a later analysis are not the required inputs for an earlier analysis.

Moreover, disciplines usually receive an unequal emphasis throughout the design process. In conceptual design, aerodynamics, propulsion, and weight estimation receive the most attention. Concerns for structures or stability receive attention during preliminary design. Finally, manufacturing and structural design considerations dominate in detail design.

2.1.4 Iterative, Sequential Process

The overall UAV design process often progresses in a manner that may be described as iterative, yet sequential (see Figure 10). Within a given design stage, disciplinary analyses may be repeated several times until the analysts feel they have arrived at a design balancing the needs of the disciplines under consideration. These iterations can account for couplings between the analyses. Thereafter, the designers proceed to the next stage and begin another iterative process considering additional disciplines at a higher level of fidelity. At this point, however, part of the design freedom has been lost, since some design variables were set in the previous stage. Moreover, since not all disciplines are considered simultaneously, the iterations will be unable to capture the complete tradespace. Additionally, disciplinary analyses are typically conducted manually, i.e. an individual enters inputs and examines results by hand. This can be a tedious and lengthy process, which individual team members are loath to repeat more than a handful of times. Hence, there is little to no motivation to ever start from the beginning and explore a different area of the design space.

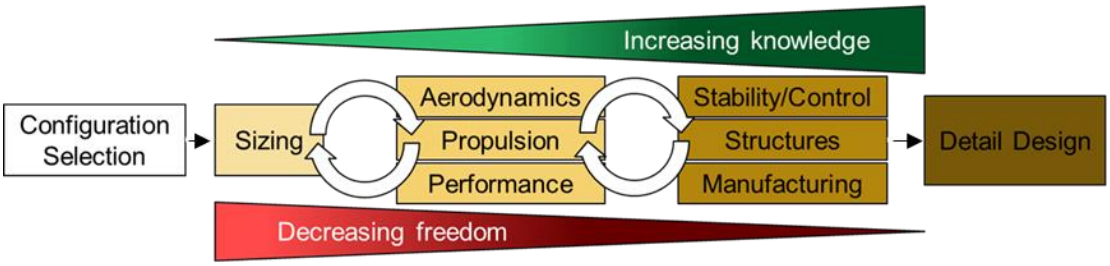

Figure 10: Traditional aircraft design process

2.2 Research Gaps

While existing UAV design methodologies have led to many successful designs, they are evidently ill-suited for configurations that exhibit strong couplings between disciplines. The unmanned electric flying wing is one such vehicle configuration, as became evident in the previous chapter. A new methodology or framework must be developed to fill the gaps that result from the strong interdisciplinary couplings inherent in this configuration. Some specific research gaps that emerged from the review are:

1. The analysis fidelity must be increased early on while there is greater design freedom.
2. For electric flying wings, disciplinary analyses need to be interlinked in a particular manner.
3. For electric flying wings, disciplinary interactions must be considered simultaneously to arrive at feasible, optimized designs.

2.3 Multidisciplinary Design Analysis

Designers have long known that vehicles as complex as aircraft invariably exhibit consequential interactions between engineering disciplines. This means that the overall system can rarely be optimized with respect to each discipline individually. Rather, the final design will be a compromise balancing the needs of structures, aerodynamics,

propulsion, stability and control, manufacturing, economics, survivability, and other concerns related to more specific vehicle requirements.

One approach for optimizing designs is to integrate discipline-specific modeling tools within a multidisciplinary design analysis. Multidisciplinary design analysis (MDA) describes approaches to engineering analysis that consider and incorporate interactions between disciplines [20]. This contrasts with traditional approaches, in which separate analysts conduct single-disciplinary analyses and set design variables without consideration for how their decisions impact other disciplines. MDA requires the integration of disparate single-disciplinary analysis codes within a larger, strongly coupled routine, in which a solution may only be identified after several convergence loops. Such integrated environments promise to yield improved designs in shorter times at lower costs, especially if they are employed in the early stages of the design process [21].

In the past, MDA techniques have been limited by their increased computational expense, which is mainly caused by the larger number of design variables. Getting disparate groups of engineers to collaborate and develop an MDA environment can also pose considerable managerial and organizational challenges. These difficulties have been partially ameliorated with advances in computing power and the availability of versatile analysis software tools such as Matlab, Excel, Mathematica, and ModelCenter.

A vital component of MDA environments is the inclusion of an interface for human interaction. Without the ability for an experienced engineer to interpret and visualize results, add constraints, and ask and then answer “what if” questions, the final design will inevitably overlook small, yet crucial considerations and lack common sense verification [20]. Visualization tools are thus just as important as analysis tools in MDA.

Some other common methodologies employed in MDA are design of experiments and approximation methods like response surface methodology and artificial neural networks. These are used together to generate surrogate models of a more computationally expensive routine. Surrogate models drastically decrease computational expense and facilitate gradient-based optimization.

In brief, MDA techniques constitute a potential gap-filler in the design of unmanned electric flying wings. MDA aims to consider interdisciplinary interactions, such as those that emerge in all-wing aircraft. Moreover, MDA effectively increases the analysis fidelity, and through an automated framework permits the exploration of a large design space early on in the design process without sacrificing design freedom. These realizations imply that a specially developed MDA framework could provide powerful improvements to the design of unmanned electric flying wings.

CHAPTER 3. PROBLEM FORMULATION

3.1 Research Question and Hypothesis Formulation

The previous chapters laid out the motivation for and challenges of designing unmanned electric flying wings. It was determined that the attributes of existing UAV design methodologies left gaps that inhibited unmanned electric flying wing design. Consequently, the central question this research effort seeks to address is:

Research Question 1.0
What methodology or framework will allow designers to overcome the challenges posed by the strong interdisciplinary couplings inherent in unmanned electric flying wings?

Following the identification of these research gaps, a potential solution was found in the field of multidisciplinary design analysis, or MDA. MDA promised to capture interdisciplinary interactions within an integrated framework and increase design knowledge without sacrificing design freedom.

Obviously, many disciplines contribute to a vehicle design project. Aerodynamics, stability, structures, weight estimation, and propulsion were already identified as particularly vital disciplines exhibiting several unique interactions on the electric flying wing configuration. These disciplines factor heavily into vehicle performance estimation, which is of immense importance, since almost all vehicles have rigid performance requirements. Additional disciplines could include manufacturing, cost estimation, and survivability. However, these other disciplines will not always be of central concern, whereas performance requirements will invariably be critical. Hence, in order to narrow

the research scope, these additional disciplines were neglected. The hypothesis that emerged in response to the research question is then:

Hypothesis 1.0
If a multidisciplinary design analysis framework is developed that interlinks aerodynamics, stability, propulsion, and weight distribution, then the challenges posed by interdisciplinary couplings on electric flying wings can be overcome.

Validating the hypothesis requires assembling disciplinary analysis tools into an integrated software environment. These tools must satisfy certain requirements to be suitable for MDA:

Speed: Analyses should execute quickly, ideally lasting no more than a few seconds.

Fidelity: Analyses should ideally be high-fidelity. Unfortunately, this goal conflicts with the previous one of fast computation, as higher fidelity tools are inevitably more expensive. In evaluating an analysis tool, one must balance the needs of fidelity and computational expense.

Automatability: The analysis tool must permit repeated execution without manual intervention. The analysis outputs must also come in an easily parsed form so that the framework software can readily pass outputs to the next disciplinary analysis.

These framework development considerations led to an additional research question, which needed to be answered for each discipline in turn:

Research Question 2.0

For each discipline, what modeling techniques will quantify the concerns relevant to unmanned electric flying wings while also satisfying the criteria of a tool suitable for MDA (speed, fidelity, and automatability)?

3.2 Roadmap

Hypothesis 1.0 supposed that by developing an MDA framework for unmanned electric flying wings, it would be possible to more rapidly identify feasible designs as compared to a traditional design process. This hypothesis can only be validated by building the proposed MDA framework and demonstrating its superior capabilities. To that end, the next chapters document the framework development, providing justifications for the modeling choices. These discipline-focused chapters address Research Question 2.0 for the core disciplines of weight estimation, propulsion, aerodynamics, and stability. CHAPTER 10 presents studies that were executed to test and evaluate the performance of the MDA framework. These results are used in CHAPTER 11 to provide a definitive response to Hypothesis 1.0.

CHAPTER 4. GENERAL IMPLEMENTATION

4.1 Requirements Definition

The first step prior to developing the MDA framework was to define the software requirements. These requirements became criteria used to evaluate and then select an appropriate development environment. The most important framework requirements are summarized below.

- Interface with external programs: The framework software needed to be capable of opening external analysis programs, issuing the analysis commands, executing the program, and parsing the generated output.
- Computational speed: Not all analyses would take place in external programs. Many computations would be conducted within the framework itself. It was desirable to accomplish these computations as rapidly as possible in order to reduce the overall analysis duration.
- Ease of development: The framework needed to be developed within a short amount of time by a single individual. Therefore, a development environment that already offered many utilities facilitating computation, data analysis, and visualization was necessary.
- Accessibility to future users: It was desirable to build an open-source framework available to any who desired to use it.
- Flexibility: A software package that the user could add to, update, and reassemble in different forms would offer much greater utility than a single, rigid integrated tool.

4.2 Development Environment Selection

Several development environments exist that could support an MDA framework. By evaluating the options qualitatively against the defined framework criteria, it was possible to select the best-suited environment.

- C++: The main attraction of C++ lies in its computational speed compared to other languages. However, developing functioning C++ programs is time intensive. C++ is also less readable and hence less flexible and accessible.
- ModelCenter: ModelCenter is a software application developed by Phoenix Integration [22]. Its enables automation and integration of external programs to solve MDA problems. Unfortunately, ModelCenter requires licensing and is therefore not universally accessible.
- Matlab: Matlab by MathWorks offers great ease of development and flexibility [23]. But compared to languages like Python or C++, its computational speed can be slow. Moreover, Matlab requires expensive licensing to use and is therefore not as accessible to future users.
- Python: Python is an open source interpreted programming language [24]. Its main features are ease of development, open accessibility, and readability. In addition, individuals and organizations have developed modules facilitating various computational and visualization capabilities. Python performs faster than Matlab in general.
- OpenMDAO: OpenMDAO is a NASA developed framework for solving multidisciplinary design analysis and optimization problems [25]. The framework is written in Python and is therefore free, openly accessible, and

relatively easy to develop. OpenMDAO provides programming classes for analysis models and formalizes input-output relations. However, this formalization restricts what the user can do within the framework.

The relative advantages and disadvantages of each potential development environment are summarized in a Pugh matrix in Table 1. Python was chosen as the baseline alternative and was ultimately selected as the programming development environment for the MDA framework due to its strong comparative metrics across all the criteria.

Table 1: Pugh matrix for framework software selection

	Python	C++	Matlab	ModelCenter	OpenMDAO
Speed	0	+	-	0	0
Ease of Development	0	-	+	0	0
Accessibility	0	0	-	-	0
Flexibility	0	-	0	0	-
Score	0	-1	-1	-1	-1

4.3 General Python Implementation

The framework was developed as a Python module. Disciplinary analyses were separated into different programming classes for modularity. Most of these classes follow a uniform programming structure.

The classes are often first instantiated by setting an “operation mode”. This operation mode determines which modeling approach is used and hence which inputs the class will accept. In this way, it becomes straightforward to add different modeling approaches in the future, assuming the inputs and outputs of physical analyses remain consistent. Following class instantiation, the user assigns model inputs to the class. Error handling is incorporated to ensure the user assigns valid values to the inputs. Then the user may execute class methods that codify a disciplinary analysis. These methods require additional simulation inputs, which define the operating condition of the model in question. To illustrate the

difference between model inputs and simulation inputs, one can take a propeller as an example. Propeller performance may be considered a function of diameter, pitch, rotational speed, and flow conditions. Within the framework, diameter and pitch are considered model parameters, while the situation-dependent rotational speed and flow conditions are simulation parameters.

Some class inputs do not define any actual model parameters, but rather inform the program where certain key files are located. For instance, aerodynamic modeling employs executable files for external analysis codes. The paths to the executables are included as class inputs.

A user may interact with the framework either through command line typing or scripting. Scripting is of course recommended for actual execution of the code. Command line typing may be helpful in developing and debugging scripts.

In brief, the MDA framework was developed as a Python programming package making extensive use of objects. The objects follow consistent structures to aid in readability. Disciplinary analyses are accessible as class methods whose execution can be readily automated in scripts, thereby permitting design space exploration.

CHAPTER 5. GEOMETRY PARAMETERIZATION

5.1 Geometry Parameterization

At the outset, it was essential to develop a geometric parameterization of the all-wing vehicle for use across all the codes. This was necessary because many disciplinary analyses require geometry as an input. To illustrate, the shape of the wing is essential for estimating lift, drag, stability, and available volume for batteries.

The chosen parameterization was adapted from the aircraft geometry modeling software OpenVSP [26]. It was selected to make interfacing with this and other aerodynamic analysis software straightforward. The parameterization assumes that the wing is symmetric about the traditional x-z plane. The wing is divided into trapezoidal sections by cutting it into slices parallel to the x-z-plane (see Figure 11). The leading and trailing edges are continuous along the span. Discontinuities could be modeled by inserting very narrow sections. Each wing section is defined by the parameters listed in Table 2 and visualized in Figure 13. The continuity constraint requires that the tip chord of one section be equal to the root chord of the next outward section. Hence, the root chord need only be defined for the innermost section.

Additional optional parameters define the size of flaps and slats on a wing section. Actuation of these control surfaces is modeled within the stability module of the framework. The parameterization requires that flaps and slats run the whole length of the section. The control surface size is defined by the ratio of its chord to the section chord at both the tip and the root of the section (see Figure 12). Hence, each flap or slat is defined by two extra parameters.

Table 2: Input parameter set for a wing section

Parameter	Units
Root chord	Length
Root airfoil	Coordinate Pairs
Root airfoil t/c scaler	Unitless
Sweep Angle	Degrees
Dihedral Angle	Degrees
Twist Angle	Degrees
Location of Twist	Fraction of root chord
Taper Ratio	Unitless
Tip airfoil	Coordinate pairs
Span	Length

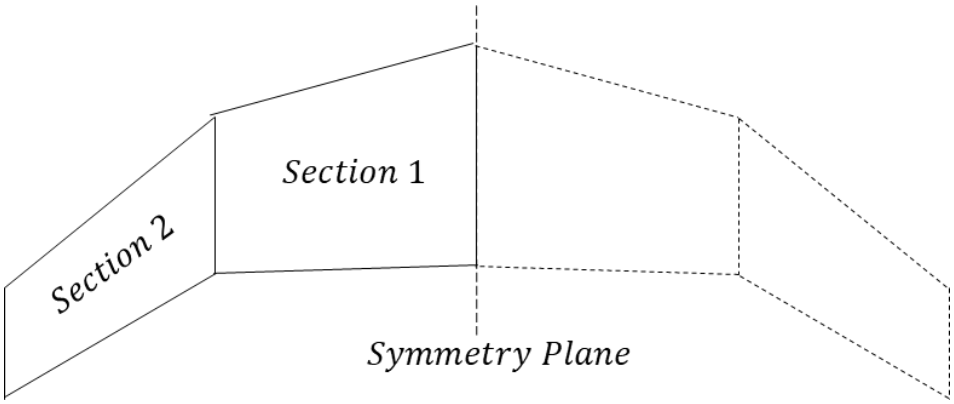

Figure 11: Division of wing geometry into spanwise sections

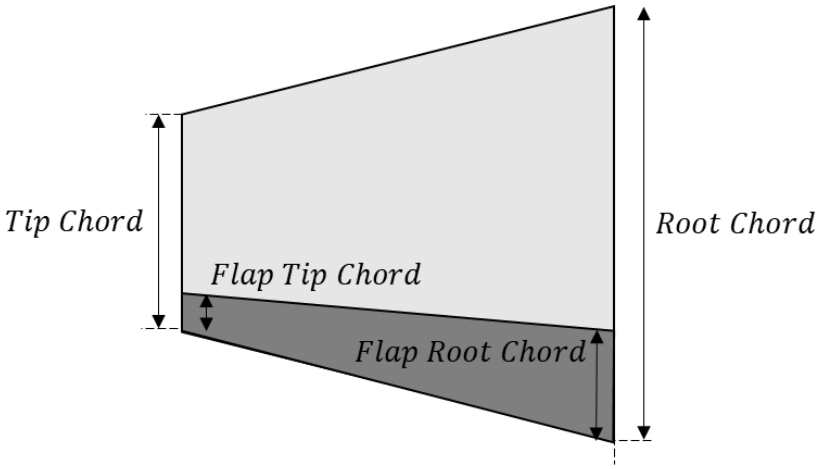

Figure 12: Flap position definition

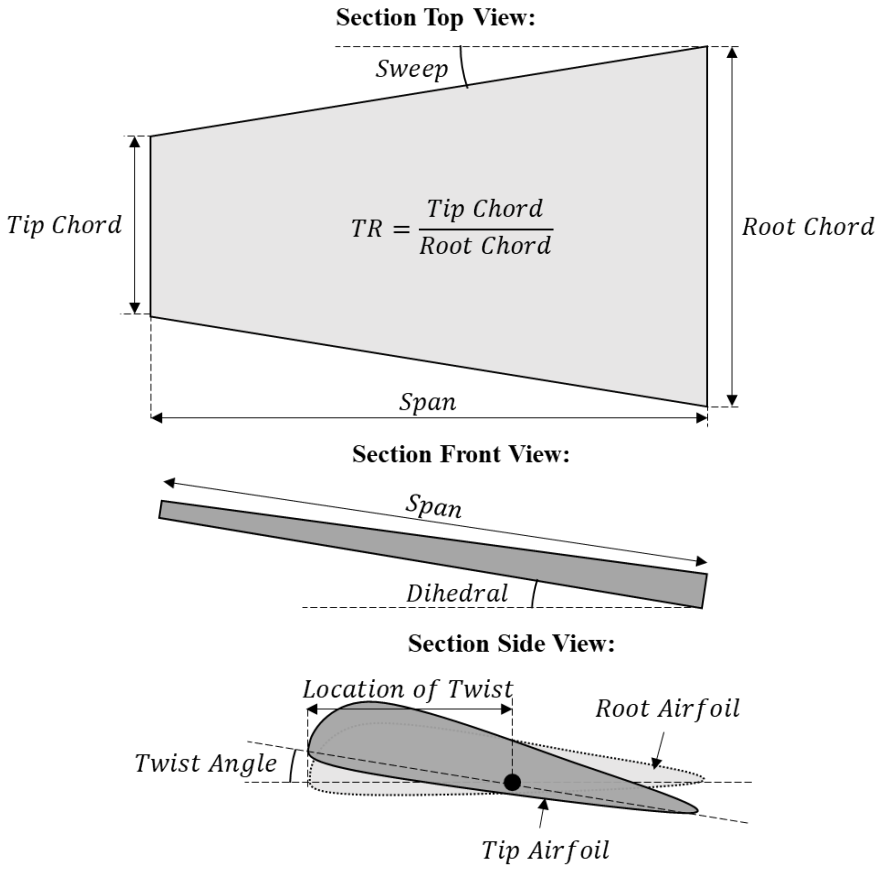

Figure 13: Wing section geometry parameterization

5.2 Geometry Module Implementation

The wing geometry was implemented as a programming class in Python. The class contains methods that allow a user to add, edit, and remove sections with ease. The sections are represented as a list of parameter key-value collections. Each collection represents a single section. A class-internal method automatically computes derived geometric parameters, such as planform area, mean aerodynamic chord, overall span, and aspect ratio. It further computes the positions of section vertices with respect to a global Cartesian coordinate system. The origin of the reference frame is located at the leading edge of the wing's root airfoil. As is shown in Figure 14, the x-axis points towards the rear of the vehicle, the y-axis points towards the starboard tip, and the z-axis points upwards. This

coordinate system was used consistently throughout the framework and is especially relevant within weight distribution modeling.

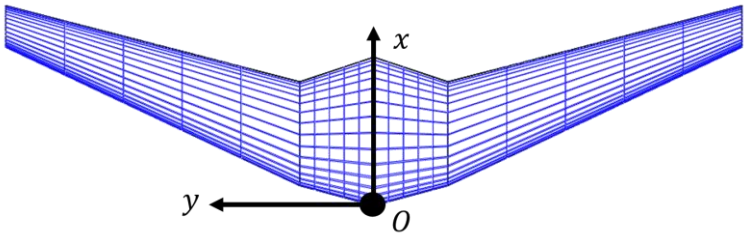

Figure 14: Example geometry exported to OpenVSP, showing global coordinate system

The geometry class has methods to automatically plot a front and overhead view of the wing. The geometry may also be scaled and stretched to achieve any desired aspect ratio or wing area (see Figure 15). When scaling, the chord and span of each section is scaled by a uniform factor. When stretching, only the span of each section is scaled by a uniform factor. Finally, a connection with the OpenVSP software makes it possible to export the geometry to a 3D model file such as IGES or STEP, while also obtaining a value for the wing volume and wetted area.

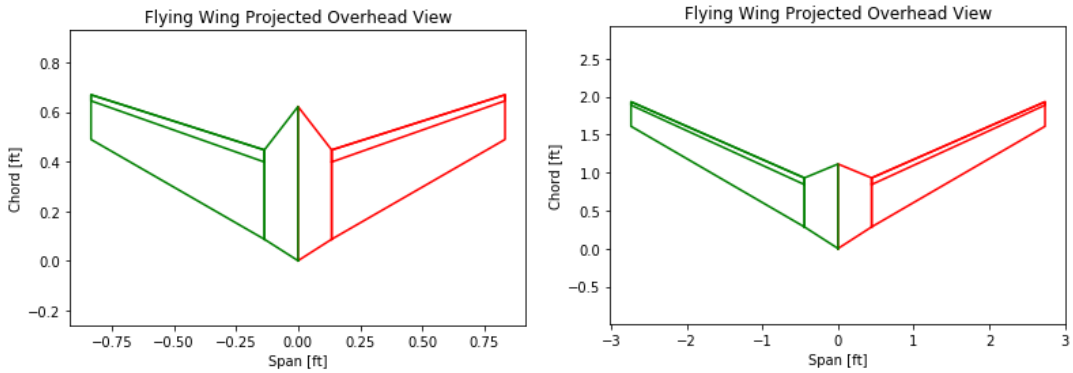

Figure 15: Geometry sketch example
(right hand geometry is a stretched and scaled version of the left-hand geometry)

In summary, the geometry module is an essential component of the MDA framework and is employed as an input in the aerodynamic and structural modeling modules that are discussed in other chapters. The implemented geometry class permits simplified definition and visualization of diverse wing shapes.

CHAPTER 6. STRUCTURES AND WEIGHT DISTRIBUTION

In simplistic terms, structural design of an aircraft aims at finding a structure that minimizes weight while satisfying requirements for stiffness and strength. The basic inputs to structural design are the aerodynamic and inertial loads on the vehicle and the geometry and material properties of the internal structure. From here, structural analysis techniques predict the internal loads and deformations. Given its evident importance in vehicle design, structures must be considered as an element within the proposed MDA framework for unmanned electric flying wings. This chapter addresses the ensuing research question:

Research Question 2.1

What modeling techniques will quantify the aspects of structural design relevant to unmanned electric flying wings while also satisfying the criteria (speed, fidelity, and automatability) of a tool suitable for MDA?

Modern structural design relies heavily on finite element analysis (FEA). FEA is used widely across industry to predict stresses and strains in structures under static, thermal, and dynamic loads. Incorporating FEA within an MDA framework would offer the highest analysis fidelity. However, FEA is computationally expensive when applied to most practical problems. Moreover, finite element solvers usually come as part of a licensed software, thereby inhibiting their open use. But more crucially, structural design requires a geometric model of the internal structure of the vehicle. Automating the generation of a structural model is possible, but not advisable for a general MDA framework. One reason for this is that structural design is an area where creativity and judgement play an essential role. Such intangibles are currently difficult to incorporate within an automated process.

Moreover, there exists such an immense variety of structural layouts, concepts, and material choices that attempting to automate all of them would be immensely laborious. Similarly, any effort to consider all discrete structural design options within an MDA framework would result in an elephantine mixed-integer problem. For these reasons, it was deemed infeasible to incorporate automated structural design and analysis within the MDA framework

Even if actual structural analysis lay outside the feasible scope of the research effort, this did not mean that structures would be neglected entirely. Ultimately, one the most important outputs of structural design is the weight and weight distribution of the vehicle. These are essential quantities to predict, as they contribute significantly to performance and stability analysis. Consequently, the strategy adopted for structures modeling within the MDA framework was to fold it into the closely related analysis domain of weight estimation. This led to the following assumption addressing Research Question 2.1:

Assumption 2.1

If weight module is developed based on a discrete mass formulation, then the most relevant cross-disciplinary impacts of structural design for unmanned electric flying wings can be captured within an MDA framework.

The remainder of this chapter documents the development and evaluation of a weight module based on this assumption. To make up for the simplifying conflation of structures and weight distribution, additional tools were developed to model feasible component placement, which is a particularly important consideration for flying wing design. This was accomplished was by developing a novel component packing algorithm used to model geometrically feasible battery cell placements within a wing. The next section motivates

the need for such a tool, and subsequent sections describe its conceptualization and implementation.

6.1 Weight Distribution Concerns on Electric Flying Wings

6.1.1 Component Placement on Flying Wings

Volumetric constraints become a critical issue in the structural design of flying wings. Since all-wing aircraft have no distinguishable fuselage, they have less overall volume in which subsystem components may be placed. Moreover, the available volume is airfoil shaped with little room towards the trailing edge. Consequently, much of the wing volume may be unusable for storing many subsystem components. Stability requirements, in particular the need to obtain a positive static margin, can further reduce the usable volume and limit placement freedom. It therefore becomes important to model valid component positionings within the wing.

Such component placement modeling for all-wing vehicles is also interesting from the standpoint of structural optimization. Recall that flying wings are advantageous because they can potentially achieve spanloading, a condition in which the weight distribution mirrors the lift distribution. Component placement could conceivably be used to achieve a spanloaded weight distribution.

Finally, knowledge of component placement contributes immense value to stability analysis. Stability modeling requires good estimates of the center of gravity and mass moment of inertia of the vehicle. Obtaining accurate estimates of these quantities is particularly important for flying wings, since static margin significantly impacts the lift distribution and stall behavior of the vehicle (this phenomenon will be discussed in greater

detail in CHAPTER 8). Component packing could offer more accurate estimates of component positions within the aircraft, thereby immensely supporting stability analysis.

6.1.2 Electric Propulsion and Weight Distribution

Electric propulsion exacerbates the component packing challenge. On an electric aircraft, the battery pack may constitute one of the heaviest single subsystem components for certain types of missions. To illustrate, the Breguet endurance equation for electric aircraft implies that endurance is maximized if the battery weight constitutes two-thirds of the takeoff weight [27] (derivation can also be found in A.7 in the appendix). A graphical illustration of this relation is provided in Figure 16. Similarly, the range can be maximized in theory if the battery constitutes the entire weight of the vehicle. Flying wings are often applied for surveillance missions requiring long endurance, meaning that optimized vehicles will likely have a large battery weight fraction. The battery weight will therefore contribute strongly to the vehicle's mass properties, and it becomes especially important to understand where within the wing the individual cells making up the battery pack could be positioned.

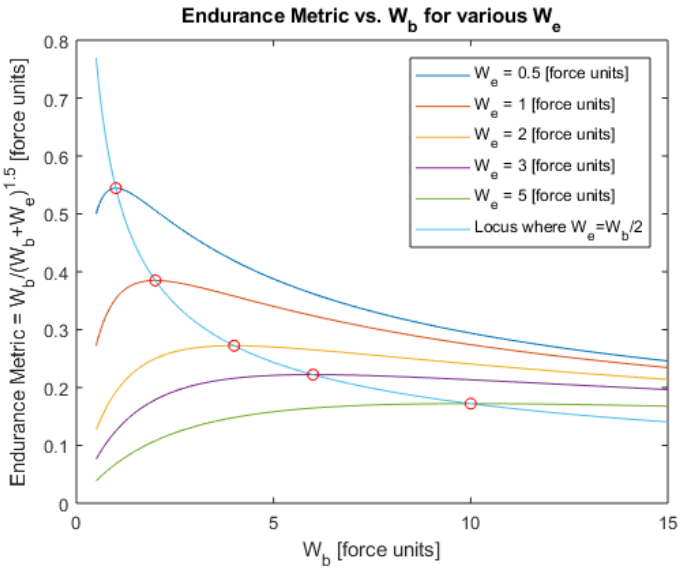

Figure 16: Theoretical dependency of electric aircraft endurance on battery weight (W_b) and empty weight (W_e).
(The endurance metric is a parameter proportional to endurance. It is assumed that takeoff weight is the sum of empty weight and battery weight.)

It should therefore be evident that battery packing has several cross-disciplinary interactions. Specifically, understanding the number and placement of batteries within a flying wing is critical to predicting:

1. The vehicle's onboard energy and power, and consequently its performance capabilities
2. The vehicle's static margin and consequently its longitudinal stability
3. The vehicle's spanwise weight distribution and consequently its structural loads and handling qualities

Any attempts at modeling the placement of components within a flying wing should therefore prioritize battery cell packing, as this information has significance across all the major disciplines. The next sections describe an original analysis routine developed to address this need.

6.2 2D Packing Algorithm

6.2.1 Conceptualization

The problem identified in the previous section was to determine feasible and dense placements of battery cells within a wing shape. Such an analysis capability would offer improved estimates of battery center of gravity, which can contribute strongly to vehicle center of gravity. The problem is 3-dimensional, but as in much exploratory research, it is valuable to approach the problem from a simpler 2D perspective first. In that case, the problem formulation becomes: Pack as many cells within an arbitrary airfoil shape as possible.

When stated in this manner, the problem becomes a cutting stock problem, which is often encountered in sheet metal cutting, textiles, and circuit board design. An overview of cutting stock problems and solution methodologies for them is included in section A.2 of the appendix. For the given problem, it is possible to make four assumptions that drastically reduce the complexity and size of the solution space:

1. Battery cells are rectangular prisms.
2. The cells to be packed are all the same shape.
3. The sides of the rectangle are either parallel or perpendicular to the chord line
4. No cell can straddle across multiple other cells

The first two assumptions can be justified by the practical realities of assembling battery packs. Lithium-ion batteries, the most energy dense commercially available types of cells, are almost invariably rectangular prismatic in shape. Moreover, one typically uses the same kinds of cells when assembling a battery pack. The other two assumptions are

made to simplify the problem and make it possible to arrive at solutions in a reasonable amount of time.

6.2.2 Implementation

In practice, one would rarely desire to pack the entire volume of a wing with batteries. Rather, one may wish only to fill a part of the wing, such as the void between two spars. To permit this flexibility, the packing area of the airfoil is defined between two bounding lines perpendicular to the chord, as pictured in Figure 17. Rectangles representing battery cells are packed between these two lines, with the airfoil curve marking the upper and lower bounds of the packing region. Six inputs are required to perform a packing routine on the airfoil:

1. Chord length c
2. Airfoil x-z-coordinates
3. x_{aft}/c (see Figure 17)
4. x_{fore}/c (see Figure 17)
5. Battery width
6. Battery height

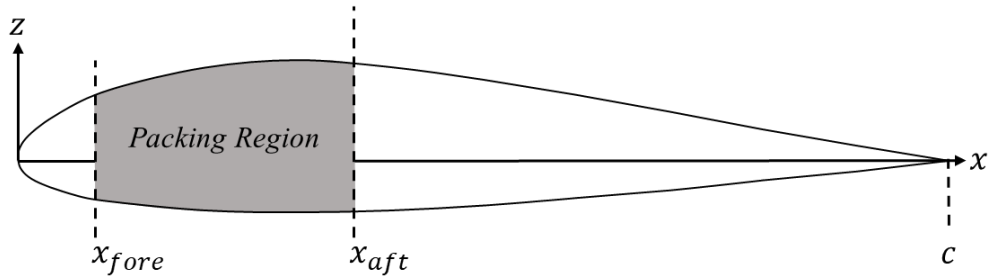

Figure 17: Packing region for the 2D packing algorithm

The selected solution methodology was heuristic, as described in section A.2 in the appendix. This was deemed the option that could provide solutions with the greatest rapidity. The algorithm performs the following steps, which will be better understood by simultaneously studying Figure 18.

1. Define $x_r = x_{aft}$.

2. Select a cell orientation. Let d_h be the dimension of the cell in the horizontal (i.e. chordwise) direction. Let d_v be the dimension in the vertical direction.
3. Check whether $x_r - d_h > x_{fore}$. In other words, check whether the cell as oriented is entirely within the packing region.
    a. If true, continue to the next step.
    b. If false, switch the values of d_h and d_v, i.e. switch the cell orientation. Recheck the condition. If it is now true, continue to the next step. If it is still false, exit.
4. Apply interpolation to the airfoil x-z-coordinates to find the z-coordinates $z_{r_{top}}$ and $z_{r_{bot}}$ of the two points on the airfoil with x-coordinates equal to x_r.
5. Apply interpolation to the airfoil x-z-coordinates to find the z-coordinates $z_{f_{top}}$ and $z_{f_{bot}}$ of the two points on the airfoil with x-coordinates equal to $x_r - d_h$.
6. Determine the number of cells than can be stacked in the interval $[x_r - d_h, x_r]$. This is equal to $\frac{\min\left(z_{r_{top}}, z_{f_{top}}\right) - \max\left(z_{r_{bot}}, z_{f_{bot}}\right)}{d_v}$. One must use integer division in this calculation.
7. Add the calculated number of stacked cells to a running tally of packed cells. Store the position of the center of area of each cell. The cells are slid down such that the lowest cell touches the airfoil contour with one of its corners.
8. Reassign x_r to equal $x_r - d_h$.
9. Return to step 2

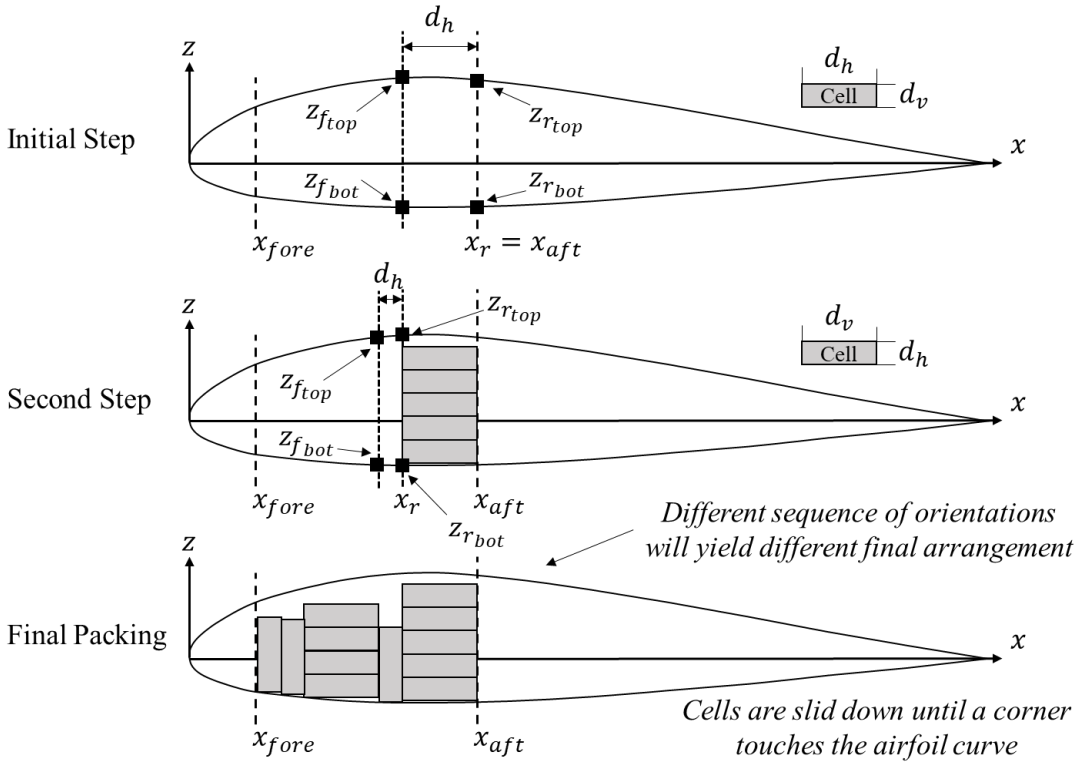

Figure 18: Illustration of the 2D packing algorithm

The ultimate packing arrangement of the cells within the airfoil depends on the sequence of cell orientations. Finding the densest packing arrangement implies performing the algorithm for each possible sequence. In the implemented routine, recursion rather than iteration was employed to check each orientation combination in a brute force manner. Recursion was chosen because it does not require explicit identification of all unique orientation combinations.

6.2.3 Evaluation

Repeated execution and visual verification of this algorithm for different inputs has demonstrated its ability to deliver feasible and dense packing arrangements of rectangles within an airfoil shape. Some sample packing solutions are presented in Figure 19, showing how changes to the cell dimensions and bounding region result in different packings. These results illustrate that the solutions make close to maximal usage of the packing space. The

algorithm may break down for airfoils with strong camber, as it does assume the airfoil contour is convex from an exterior perspective.

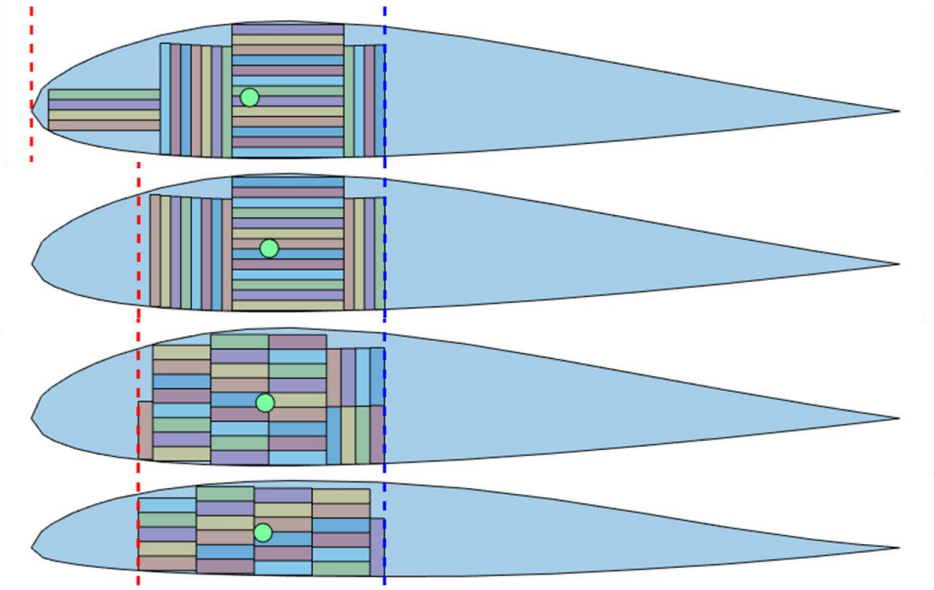

Figure 19: Example 2D packing solutions

6.3 3D Packing Algorithm

6.3.1 Conceptualization

The prior section described the development of an algorithm to pack rectangles within a 2D airfoil. The real problem at hand, however, was packing 3D cuboids within a 3D wing shape. Even with the assumption that all cuboids (i.e. battery cells) have the same dimensions, the densest packing would likely result from placing the cells at different orientations and permitting straddling of cells over one another. In this case, a hybridized heuristic-metaheuristic approach seems apt (refer to A.2 in the appendix for an explanation of this methodology). However, developing such an optimization routine would require a considerable amount of time investment and the execution duration would likely be too large for use within an MDA framework. Moreover, the goal of the endeavor was not to find the densest possible packing, but rather to find *feasible* packings that were *reasonably* dense in order to better estimate the weight distribution of batteries. By continuing with

the assumptions that cells do not straddle each other and can only take on two orientations, the fastest and simplest solution to the 3D packing problem was to employ the already developed 2D heuristic algorithm.

In simple terms, the approach taken was to divide the wing into smaller spanwise segments of lengths equal to the length of the battery cell. Each segment could be approximated as an extruded airfoil without any taper or twist. This means one could apply the 2D packing algorithm to one end of the segment and obtain a dense arrangement of rectangles. Extruding these rectangles produced a feasible 3D cuboid arrangement within the wing segment. Redoing this analysis on each segment yielded the overall packing.

The assumption of no twist may appear very limiting, given that sweptback flying wings must include washout to be stable. However, if the twist is gradual over the length of the span, or the length of the cuboids is sufficiently small, then the predicted maximum number of packable cuboids remains valid.

6.3.2 Implementation

The first step in developing the 3D packing algorithm was the definition of required inputs. Just as in the 2D packing algorithm, it was desirable to restrict the region in the wing that was to be filled in. This was accomplished by defining a trapezoidal region through six dimensionless parameters, $c_{f_i}, c_{f_o}, c_{a_i}, c_{a_o}, s_{start}$, and s_{end}. The meaning of these parameters can be best understood by examining Figure 20.

Additional inputs included the dimensions of the cells and the airfoil point coordinates for the root and tip of the wing section. The dihedral, sweep, taper ratio, root chord, and position of the root leading edge in the global reference frame were also necessary to accurately calculate the position of each cell.

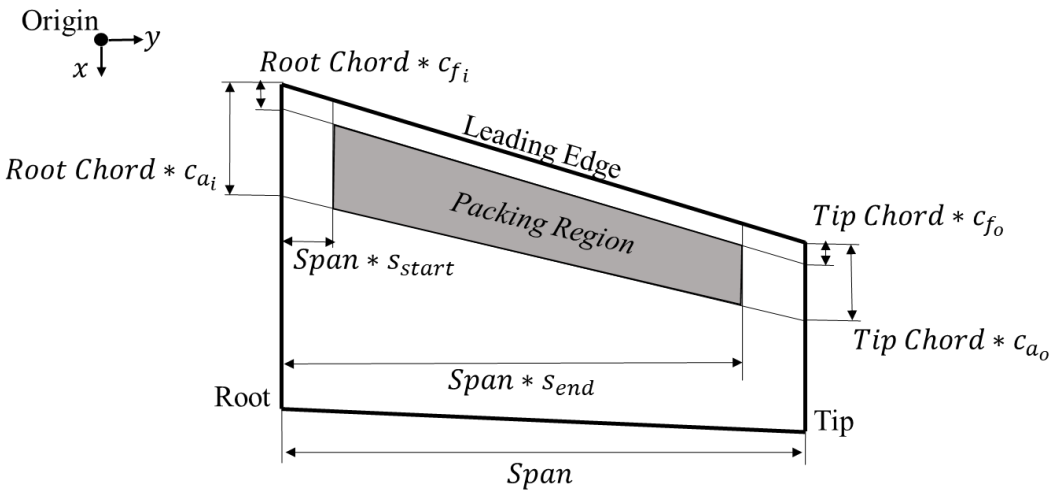

Figure 20: Definition of the 3D packing region

The wing section was represented using two airfoils defined by discrete x-z-pairs. The x/c values of all the points needed to be consistent for both airfoils. The airfoils were transformed into the 3D space of the global reference frame to represent the root and the tip of the wing section. Lines connecting points at the root and tip with corresponding x/c values represented the exterior surface. These lines, also pictured in Figure 21, are referred to as the mold lines.

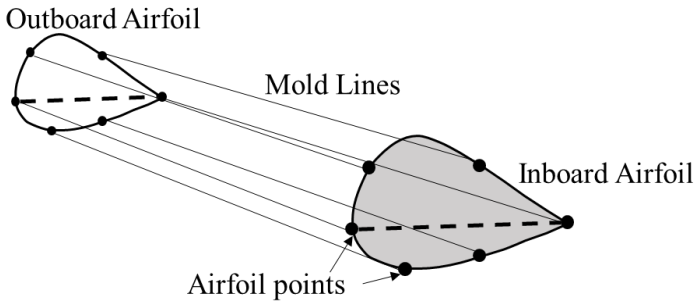

Figure 21: Wing surface representation using mold lines

The 3D algorithm follows the sequence of steps described below. Figure 22 provides graphical definitions of the parameters to which these steps refer. It assumes that the wing tapers off in the outboard direction. A modification to the code would be required to properly consider the rare cases where the taper ratio is greater than 1.

1. Identify the coordinates of the points $\vec{r}_{a_i}$, $\vec{r}_{a_o}$, $\vec{r}_{f_i}$, and $\vec{r}_{f_o}$ using the input parameters shown in Figure 20. These points lie on a plane connecting the root and tip chord lines. Such a plane exists because it is assumed there is no twist.
2. Define the vector $\hat{n} = \frac{\vec{r}_{a_i} - \vec{r}_{a_o}}{\left\| \vec{r}_{a_i} - \vec{r}_{a_o} \right\|}$.
3. Define the initial starting point $\vec{r}_s$
    a. If the wing section has forward sweep, then $\vec{r}_s = \vec{r}_{f_i} - l\hat{n}$. l is the length of the battery cells.
    b. If the wing section has backward sweep, then $\vec{r}_s = \vec{r}_{a_i} - l\hat{n}$.
4. Define a plane p passing through the point $\vec{r}_s$ normal to $\hat{n}$.
5. Find the intersection points of p with the lines $\overline{\vec{r}_{a_i}\vec{r}_{a_o}}$ and $\overline{\vec{r}_{f_i}\vec{r}_{f_o}}$. The former point will of course simply be $\vec{r}_s$. The distances of these points along the chord from the leading edge correspond to x_{aft} and x_{fore} in the 2D packing algorithm.
6. Check that the two intersection points in fact lie on, rather than beyond, the line segments $\overline{\vec{r}_{a_i}\vec{r}_{a_o}}$ and $\overline{\vec{r}_{f_i}\vec{r}_{f_o}}$ respectively.
    a. If true for both, continue to the next step.
    b. If false for either, exit.
7. Find the coordinates of the intersection points of p with the mold lines. These are the coordinates defining the intersection airfoil used in the 2D packing.
8. Transform the intersection airfoil coordinates into a 2D space and conduct the 2D packing algorithm.

9. Transform the centroids of each packed rectangle ($\vec{r}_{centroid_{rectangle}}$) back into the global reference frame.
10. Compute and save the centers of gravity and moments of inertia of extrusions of the packed rectangles. The rectangles are extruded in the direction $\hat{n}$ by the length l. The resulting center of gravity of a cuboid $\vec{r}_{cg_{cuboid}}$ is given by Eqn. (1). The mass moment of inertia tensor for each cell is first computed using the standard equations for a cuboid before being transformed into the global reference frame.
11. Reassign the starting point: $\vec{r}_s = \vec{r}_s - l\hat{n}$.
12. Return to Step 4.

$$\vec{r}_{cg_{cuboid}} = \vec{r}_{centroid_{rectangle}} + \frac{l}{2}\hat{n} \tag{1}$$

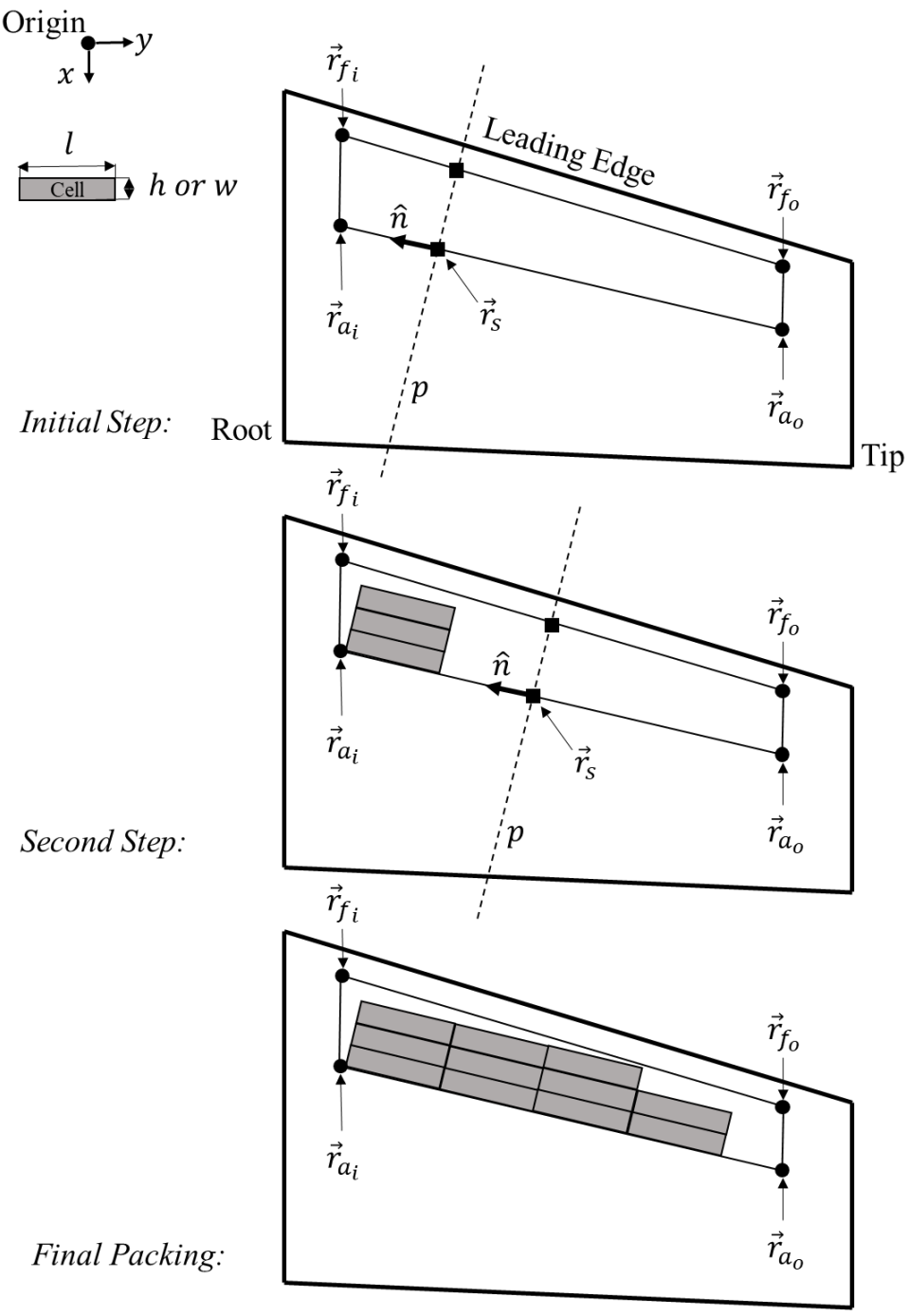

Figure 22: Illustration of the 3D packing algorithm

6.3.3 Evaluation

Visualizations of the 3D packing results, such as that shown in Figure 23, indicated that the solutions left very little space for any additional cells to be added. These tests were taken as verification that the algorithm supplied consistently reasonable and usable results.

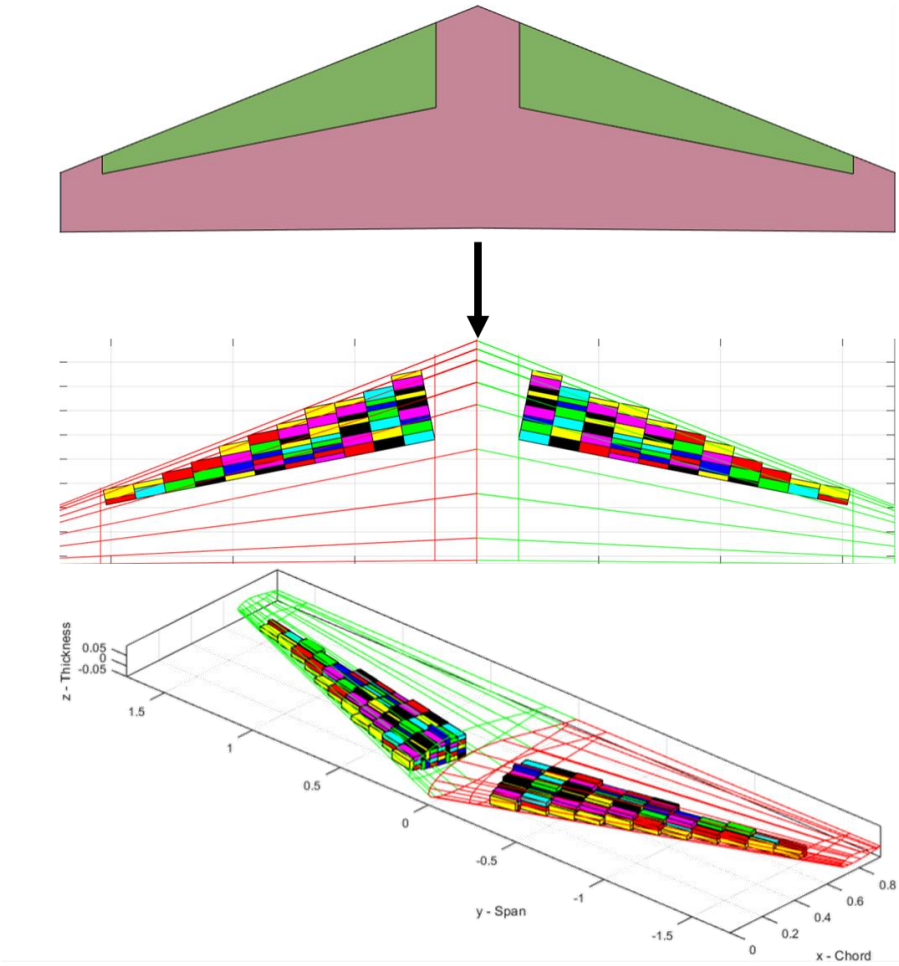

Figure 23: 3D packing solution for a swetpack, tapered wing
(The green region in the upper planform sketch represents the packing region. The lower two visuals show the packing inside the wing shape from two vantage points.)

A notable deficit of the algorithm is that it assumes the internal volume of the wing is completely empty, devoid of any spars, ribs or other internal structure. Only skin thickness can be easily modeled by shrinking the mold lines of the wing. At first glance, this appears to be a severely limiting and unrealistic assumption. However, the tool still offers valuable insights by providing an upper limit on the packing capacity within the wing. Consequently, the 3D and 2D packing algorithms represent important contributions to weight and weight distribution modeling for electric flying wings or any other application area where one wishes to determine feasible, dense packing arrangements of cuboids within a curved, extruded body.

6.4 Structures Module Implementation

6.4.1 Mass Tabulation

The structures and weight distribution module was implemented as a Python programming class. The primary method in the class enables a user to add, remove, replace, or edit individual mass components. These components are stored as rows in a class-internal table, which may be output to a file for separate examination. Each component is a vector of relevant information, including a unique name, the mass, the location of the center of gravity with respect to the origin, and the elements of the mass moment of inertia tensor:

$$Component_i = [name_i, description_i, m_i, x_{cg_i}, y_{cg_i}, z_{cg_i}, I_{xx_i}, I_{yy_i}, I_{zz_i}, I_{xy_i}, I_{xz_i}, I_{yz_i}]$$

In many situations, the exact center of gravity or moment of inertia tensor of a component may not be known. These are therefore optional inputs. If no information is provided, the moments and products of inertia are equal to zero, and the component's mass will not be considered in the center of gravity calculation.

Any time a change is made to the internal mass table, the module automatically recalculates the mass, center of gravity, and moment of inertia tensor of the entire vehicle. Given *N* components, vehicle mass m_{veh} is computed using Eqn. (2). The center of gravity vector $\vec{r}_{cg}$is then determined by Eqn. (3). Once $\vec{r}_{cg}$ is known, mass moments and products of inertia are calculated using Eqn. (4) through Eqn. (10).

$$m_{veh} = \sum_{i=1}^{N} m_i \tag{2}$$

$$\vec{r}_{cg} = \frac{\sum_{i=1}^{N} m_i \vec{r}_{cg_i}}{m_{veh}};\ \vec{r}_{cg_i} = \begin{bmatrix} x_{cg_i} \\ y_{cg_i} \\ z_{cg_i} \end{bmatrix} \tag{3}$$

$$I_{xx} = \sum_{i=1}^{N} \left[(y_i^2 + z_i^2) m_i + I_{xx_i} \right] \tag{4}$$

$$I_{yy} = \sum_{i=1}^{N} \left[(x_i^2 + z_i^2) m_i + I_{yy_i} \right] \tag{5}$$

$$I_{zz} = \sum_{i=1}^{N} \left[(x_i^2 + y_i^2) m_i + I_{zz_i} \right] \tag{6}$$

$$I_{xy} = I_{yx} = -\sum_{i=1}^{N} \left[x_i y_i m_i + I_{xy_i} \right] \tag{7}$$

$$I_{xz} = I_{xz} = -\sum_{i=1}^{N} \left[x_i z_i m_i + I_{xz_i} \right] \tag{8}$$

$$I_{yz} = I_{zy} = -\sum_{i=1}^{N} \left[y_i z_i m_i + I_{yz_i} \right] \tag{9}$$

$$\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \vec{r}_{cg_i} - \vec{r}_{cg} \tag{10}$$

6.4.2 Structural Mass Estimation

The structures class may be operated in two different "operating modes" depending on how structural mass is being estimated. The first mode estimates the aircraft structure as a monocoque shell with thickness t_{shell} and density ρ_{shell}. The area of the shell is equal to the wetted area of the vehicle S_{wet}, which can be determined by OpenVSP. The structural mass of the vehicle m_{struct} in this mode is calculated using Eqn. (11), which assumes the skin thickness is very small.

$$m_{struct} = S_{wet}\, \rho_{shell}\, t_{shell} \tag{11}$$

Alternatively, structural mass can be modeled as a fraction of the total takeoff weight using a second operating mode. In this case, the structural mass is computed from Eqn.

(12), where MF_{struct} is an input parameter equal to the ratio of structural mass to total mass. The term $\sum_{i=1}^{N-1} m_i$ is the mass of all components excepting the structural mass itself.

$$m_{struct} = \frac{\sum_{i=1}^{N-1} m_i}{1 - MF_{struct}} \tag{12}$$

In either case, the center of gravity of structural mass is placed at the geometric centroid of the wing planform by default. In calculating this centroid, the wing is treated as a set of trapezoidal areas with no thickness. The centroid is evaluated by implementing Eqn. (13) on this simplified wing model, where $\vec{r}_c$ represents the centroid of the elemental area dA. A potential future improvement would be to give greater weight to the areas further inboard, as the structural mass would likely be larger in those regions.

$$\vec{r}_{cg_{struct}} = \frac{\int \vec{r}_c \, dA}{\int dA} \tag{13}$$

6.4.3 Cuboid Packing

The weight distribution module's final realm of functionality relates to cuboid, i.e. battery cell packing. A user may declare an arbitrary number of packing regions within the wing. These regions can be edited or removed subsequently. Each region is defined by the index of the wing section on which it is placed, the dimensions of the cuboid to be packed, and the parameters defining the bounds of the region. Each packing region is mirrored on both sides of the wing. Further inputs are the minimum and maximum number of cuboids that must be filled within the mirrored regions. To model battery cells specifically, an additional integer input stating the number of cells in series is available. The program can then ensure that the total number of cells packed into the regions equals an integer multiple of this number.

In addition, the module provides a simple method of accounting for volumetric losses due to skin thickness. The user can set a volume shrink factor for the analysis, which will virtually scale down tip and root chords of the wing sections that are input to the 3D packing algorithm. In this manner, the mold lines passed to the algorithm represent the interior volume of the wing.

After all packing regions have been defined, the user may call the 3D packing algorithm to determine the densest packing arrangement for each region. Failing to pack the minimum number of cuboids in a region leads to an error. Any additional cells in excess of the maximum permitted number are deleted automatically. In such a situation, the cells furthest aft are removed first in order to shift the center of gravity further forward. Finally, each cell is added to the internal mass property table.

Execution time for the packing code depends on the relative sizes of the wing and cuboids: Packing small cuboids within a large packing area requires more computational effort and time, as there are more unique orientation combinations that need to be compared. But the runtime for most cases is still on the order of seconds. For instance, the packing depicted in Figure 24 took 1.05 seconds to evaluate.

6.4.4 Visualization Support

In order to verify the packing regions, the user can output the wing geometry with the batteries to a 3D model using the software OpenVSP [26]. These visualizations have been used to qualitatively verify the results of the 3D packing algorithm. An example visualization is included in Figure 24, which shows that the effect of dihedral is properly considered.

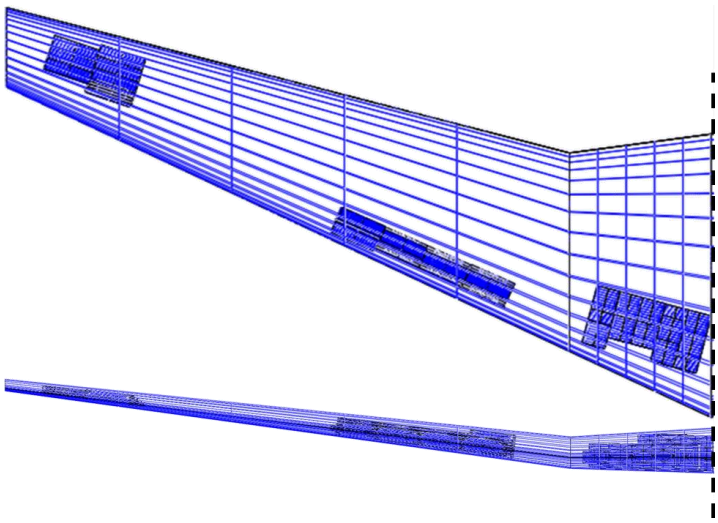

Figure 24: OpenVSP packing visualization for one half of a wing with three separate packing regions

A further form of visual interaction with the structures module is provided in the form of an overhead mass distribution plot, as seen in Figure 25. This method sketches an overhead view of the wing and draws circles at the location of each mass component, with the radius of each circle suggesting the relative magnitude of the mass. This plot provides the user with an additional diagnostic for verifying that the mass distribution is indeed defined as intended.

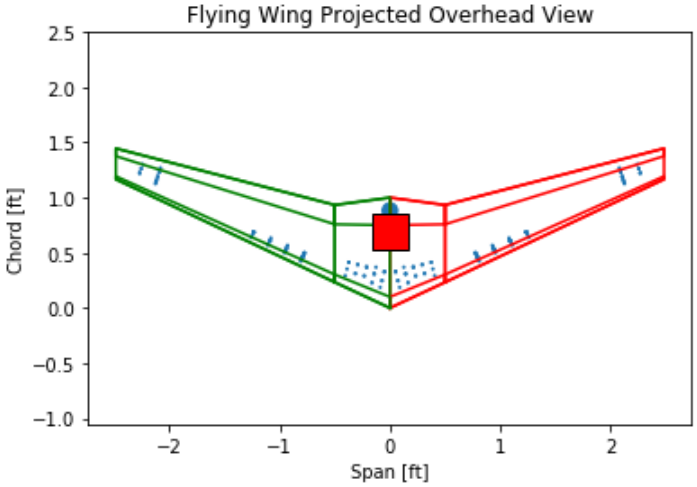

Figure 25: Example overhead mass distribution plot
(dots represent component masses; red square represents center of mass)

6.5 Weight Module Summary

In brief, this chapter has addressed the issue of structural modeling of unmanned electric flying wings within an MDA framework. At the outset, it was determined that high fidelity structural analysis would be inappropriate to include within a general MDA framework due to its computational expense and need for a specific geometry model. Consequently, it was decided to conflate structures modeling with weight estimation. Feasibility of component placement was identified as a major issue for the flying wing configuration, and battery cells emerged as the component for which placement determination was most crucial. This led to the desire to develop a routine to predict dense, feasible packings of battery cells within wing structures.

A 3D packing algorithm was presented that could place cuboids representing battery cells in dense arrangements within an airfoil extrusion. This code represents perhaps the most original contribution developed as part of this thesis and could be used in other applications where it is necessary to densely pack rectangular shapes within a curved shape.

The packing algorithm was included as only a portion of a larger weight distribution module. This module's primary function was to compute mass, center of mass, and mass moment of inertia tensor using a discrete mass formulation. Structural mass was estimated as either a monocoque shell or a fraction of total mass. The module further provided visualization capabilities to verify packing arrangements.

In future, this module could be improved to provide some basic structural analysis to roughly estimate such things as maximum bending stress or tip deflection. But in its current state, it is still capable of outputting the most crucial cross-disciplinary structures-related parameters, namely weight and weight distribution. Moreover, it can model the placement

of batteries and other cuboid-shaped components within the vehicle, thereby offering a higher level of accuracy in the modeling environment than the alternative of simply guessing component locations without regard for their geometric feasibility. This demonstrated capability indicates that the developed module considers the interaction between structural design and weight distribution estimation to a higher degree of fidelity than is commonly the case in early design. Consequently, the module captures the most relevant cross-disciplinary impact of structural design to a satisfying degree.

CHAPTER 7. ELECTRIC PROPULSION

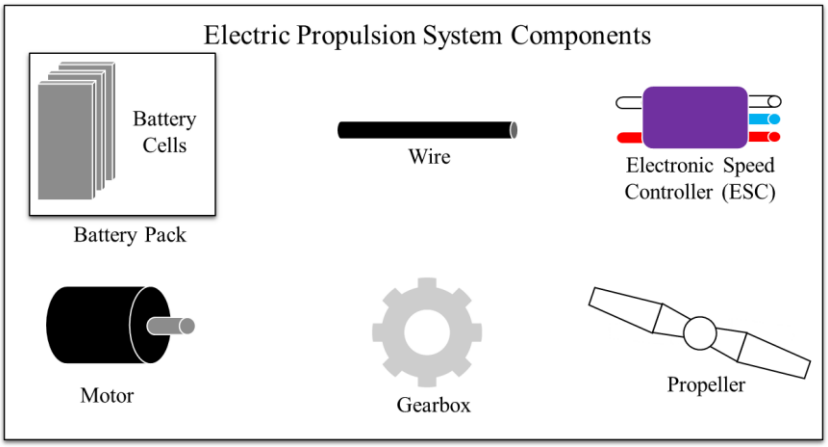

Figure 26: Components of an electric propulsion system for UAVs

A basic electric propulsion system consists of a battery, wires, a speed controller, a motor, a gearbox, and finally a propeller or ducted fan (see Figure 26). Propulsion modeling is essential to performance analysis, since it predicts the power input and power output at an operating condition. In addition, the propulsion system components contribute weight to the vehicle, and their positioning affects stability characteristics. Finally, some externally visible portions of the system, such as ducts or propellers, may alter vehicle aerodynamics. These various interdisciplinary effects indicated that the electric propulsion system was an essential module within a multidisciplinary design analysis framework for unmanned electric flying wings. This conclusion led to the following research question:

Research Question 2.2:
What modeling techniques will quantify the performance, weight, and aerodynamic impacts of an electric propulsion system while also satisfying the criteria (speed, fidelity, and automatability) of a tool suitable for MDA?

Some added background review helped provide an initial answer to this question. First, the primary interaction between propulsion and performance comes in the form of estimating power consumption at an operating condition. Power losses are associated with each step in the transmission process from battery to propeller. Consequently, a propulsion module must include power loss models for each system component if it is meant to affect performance estimation in any meaningful way.

The propulsion system's impact on weight estimation can be readily captured by tabulating the individual weights and estimated positions of each component. Data-based mass regressions or analytical expressions may be employed in cases where exact component masses are not known.

By contrast, the aerodynamic impacts of externally visible components such as ducts and propellers are difficult to predict in general without recourse to computationally intensive analysis. A simple way to model the effect would be to add a constant contribution to drag coefficient. But aside from including this calibration knob, modeling the aerodynamic impacts of the propulsion system was considered outside the present research scope.

Taking this additional background review into account, it was possible to formulate an assumption addressing research question 2.2:

Assumption 2.2:

If a propulsion module including component power loss and mass models is developed, then the interdisciplinary effects of electric propulsion can be quantified within an MDA framework.

The remainder of the chapter discusses the development and testing of a model code based on this assumption.

7.1 Existing Modeling Tools

The simplest modeling approach for electric propulsion is to assume constant component efficiencies. However, such a simple assumption is wholly inadequate when moving beyond back-of-the-hand calculations. Higher fidelity system models consider the dependence of component efficiencies on an operating condition.

Two examples of such higher-fidelity routines are MotoCalc and eCalc. MotoCalc and eCalc are software programs for analyzing electric power systems and fixed-wing vehicle performance [28] [29]. They are mainly used by hobbyists. Both tools predict power system performance, outputting amongst other things the current, voltage, motor RPM, and propeller thrust. MotoCalc can provide these predictions for various airspeeds and throttle settings and has proven to be a good tool for estimating drive system performance, although verification through physical testing should never be neglected [30]. Unfortunately, both programs require a payed license. In addition, they are only accessible through a graphical user interface, which is difficult to automate. This makes it difficult to incorporate either tool within an MDA environment.

Since existing tools would be difficult to incorporate within an MDA framework, it became necessary to develop a model from the ground up. To accomplish this, it was necessary to understand and then model each component of the power system. The following sections address each system component in turn, documenting background information and model implementation.

7.2 Propeller

The propeller is the almost universally employed means of providing thrust on low-speed aircraft. It may take the form of either an open rotor or a ducted fan. The functional principles of propellers are the same as those of wings (see Figure 27). As the propeller rotates within a flow field, fluid passes over an airfoil-shaped section, producing lift and drag. These forces can be decomposed into components parallel and perpendicular to the axis of rotation. The parallel component results in the thrust. The perpendicular component results in a force resisting the rotation of the propeller. This creates a torque that must be overcome for the propeller to rotate at the given rate. The thrust and torque quantify the performance and efficiency of the propeller at an operating condition

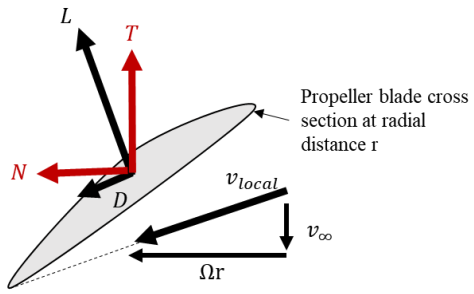

Figure 27: Forces on a propeller section

(T and N sum to the same force as L and D. Ω is the rotational rate. v_∞ is the free stream velocity.)

7.2.1 Propeller Performance Analysis Methods

There are two major methods to incorporate propeller performance modeling within an MDA framework. The first is to query stored performance maps. The second method is to automate a propeller analysis tool. Both approaches were implemented within the propulsion module for enhanced flexibility.

Performance Maps

In the performance map approach, experimental or analytical performance data is gathered into a table, which a computer code can then query to obtain the propeller

performance at a desired operating condition. This approach is fastest for obtaining performance estimates within an MDA framework. However, it presupposes the existence of an accurate map. Two potential sources for propeller maps are the experimentally gathered UIUC propeller database and the analytically computed performance data of APC brand propellers [31] [32]. However, the former source has a restricted data domain, while the latter has questionable accuracy. A further disadvantage of the map approach is that the Reynolds number and Mach number of the desired operating condition will usually not match the values at which the performance map data was obtained.

For the sake of modeling flexibility, the developed propeller model code allows a user to connect a properly formatted propeller map file. This file tabulates thrust and torque values at different airspeeds and rotational rates. The code then employs table interpolation to evaluate the thrust and torque at an arbitrary airspeed and rotational rate. A numerical solver is employed to invert the table lookup. This capability is necessary when, for instance, the thrust, rather than rotational rate, is prescribed. The model does not permit extrapolation beyond the domain of the performance map.

Analytical Tool Automation

The analysis automation approach requires an automation wrapper that passes inputs to a propeller analysis tool, executes the tool, and then parses the outputs. The analysis tool needed to be both fast and publicly available. These requirements ruled out computational fluid dynamics software. Instead, three open-source propeller analysis tools relying on blade element vortex theory were considered: Qprop, Xrotor, and DFDC.

Qprop is a combined motor-propeller analysis tool developed by Harold Youngren and Mark Drela [33]. Given an input file containing a propeller's geometry and sectional

aerodynamic characteristics, Qprop can provide predictions of torque and thrust for a given rotation rate and airflow. The blade section aerodynamic properties can be estimated using a 2D flow solver such as Xfoil [34].

Xrotor is another Drela-Youngren software product, which uses the same theory in Qprop but offers routines to design optimal propeller blades for a given operating condition [35] [36]. Xrotor has the added capability of modeling ducted fans to a superficial degree.

Ducted Fan Design Code (DFDC) is a third Drela-Youngren product which provides higher fidelity modeling routines for ducted fans [37]. A DFDC input file requires information not only about the propeller but also about the duct geometry.

Qprop, Xrotor, and DFDC execute quickly, and since they are run from the computer command line, they can be easily automated with Python's subprocess module. This, together with the fact that all three programs are publicly available, makes them ideally suited for incorporation within an MDA framework for unmanned electric flying wings. Within the framework, only an Xrotor automation wrapper was implemented. A DFDC wrapper could be developed in future. There was little point in including Qprop, as Xrotor contains the same propeller formulation, while also being able to model ducted fans.

7.2.2 Parametric Propeller Model Development

One of the disadvantages of using maps or individual Xrotor input files for performance analysis is that the results only apply for a single propeller. Multiple propellers could be considered by employing several maps within a design space exploration, but this would result in a discrete rather than a continuous design space. In order to effectively examine sensitivities and trade-offs, it was desirable to model propeller

performance in terms design variables like diameter and pitch. This was accomplished within the propulsion module through an original parametric propeller model.

The parametric propeller model takes in a small set of inputs:

- Diameter
- Pitch
- Style
- Blade Airfoil
- Solidity Scaling Factor
- Number of Blades
- Hub Radius
- Material

From these inputs, an Xrotor input file is automatically generated which may then be passed to an Xrotor analysis. Propeller mass is also estimated using these parameters and a regression developed by Bershadsky [38]. More details on the implementation of the parametric propeller model are provided in section A.3 in the appendix. Overall, the model represents a valuable capability within the framework, as it eliminates the need for predefined propeller maps or Xrotor input files. Moreover, it can be used to quantify the sensitivity of vehicle performance to propeller design variables.

7.3 Gearbox

7.3.1 Gearbox Background

In the past, electric motors tended to operate most efficiently at higher rotational rates than those at which a propeller would achieve its maximum efficiency. As a result, gearboxes were used to transform the high-speed, low-torque rotational power of the motor to low-speed, high-torque power for the propeller. Unfortunately, gearboxes add weight and power losses due to friction and noise in the gears. This power loss needs to be considered when estimating the performance of the propulsion system.

More recently, motor manufacturers have developed low-speed motors for the small UAV market. Such motors are characterized by low speed constants (K_v) and higher output torques, making them especially suited for driving propellers. As a result, even though

gearboxes should be included within the MDA framework for flexibility and completeness, they are no longer a significant concern in electric UAV propulsion modeling.

7.3.2 Gearbox Model Implementation

The gearbox model implementation was straightforward. It is defined in terms of only mass, gear ratio (k_{GB}), and efficiency (η_{GB}). Efficiency is assumed to be constant. Typical values range between 0.9 and 1.0 [28].

Two model functions were implemented. One function computes the motor torque and rotational rate based on the propeller torque and rotational rate. The second function does the inverse. A no-slip condition in the gears is assumed. Hence, the efficiency is presumed to only impact the torque transformation. The transformation equations from motor side to propeller side are given in equations (14) and (15):

$$\Omega_p = \Omega_m / k_{GB} \tag{14}$$

$$Q_p = Q_m k_{GB} \eta_{GB} \tag{15}$$

Ω_m and Ω_p are the rotational rates of the motor and propeller respectively. Similarly, Q_m and Q_p are the torques of the motor and propeller. The inverse functions are easily obtained by rearranging the equations.

7.4 Electric Motor

7.4.1 Motor Background

The electric motor converts electrical power into mechanical power. “Brushed” motors are supplied by direct current, while “brushless” motors use three-phase current. The torque produced by the motor is roughly proportional to the current, while the speed is roughly proportional to the voltage. The approximate constant of proportionality is commonly quoted in RPM/V and called the “K_v” of the motor. Motors introduce power losses through

a variety of mechanisms, including noise and friction. Two prominent sources of inefficiency are (1) the resistance of the wire coils and (2) the fact that there exists a no-load current in the motor, i.e. even when the motor produces negligible torque, a nonzero current flows through it. The motor may additionally be defined by its mass and maximum rated current flow. If the current exceeds this maximum rating, then the wires are in danger of heating to the point of fusing together, at which point the motor is unusable.

7.4.2 Motor Model Implementation

Overall, it is possible to obtain a first-order motor power loss model using three parameters: K_v, resistance R_m, and no-load current I_0. Conveniently, motor datasheets commonly quote these three values, making it straightforward to model a motor without conducting benchtop experiments to determine model parameters. More information on motor modeling can be found in the documentation for Qprop [39].

As implemented, the motor model provides one primary functional method: the computation of motor efficiency as a function of rotational rate and applied torque. The equations for this function were adapted from [39]. The first step is to compute the equivalent DC current I_m passing through the motor, as shown in Eqn. (16).

$$I_m = K_v \, Q_m + I_0 \tag{16}$$

Motor efficiency follows from Eqn. (17)

$$\eta_m = \frac{1 - I_0/I_m}{1 + I_m \, R_m \, K_v/\Omega_m} \tag{17}$$

The no-load current and coil resistance will usually be the value quoted by the motor manufacturer. But if a certain motor is available for further bench testing, additional model parameters accounting for nonlinear behaviors may be estimated and included. As

documented in Eqn. (18) and Eqn. (19), coil resistance may be modeled as a quadratic function of current, and no-load current as a quadratic function of rotational rate:

$$R = R(I_m) = R_0 + R_2 I_m^2 \tag{18}$$

$$I_0 = I_0(\Omega_m) = i_0 + i_1 \Omega_m + i_2 \Omega_m^2 \tag{19}$$

Experimental methods by which these added model parameters (R_0, R_2, i_0, i_1, i_2) may be estimated are described in [33]. Note that if R_0 is set to the nominal R_m, and i_0 is set to the nominal I_0, and the remaining parameters are all set to 0, then the lower-order motor model falls out. It is typically not possible to estimate these higher order parameters during initial design studies aimed at exploring different motor options. Assumptions of constant coil resistance and no-load current would have to suffice in such situations.

7.4.3 Reduced Order Motor Model

In most UAV design projects, it is common to select commercially available motors, rather than designing a motor to a certain specification. Hence, it is likely impossible to find a motor satisfying any desired values of K_v, R_m, I_0, mass, and maximum rated current I_{max} simultaneously. In practice, it is usually only possible to select a motor satisfying two parameter values. This realization leads to the possibility of eliminating degrees of freedom within the motor model and reducing the number of necessary inputs, thereby simplifying design space exploration.

Of the five parameters, K_v and I_{max} will usually be the most important to set. I_{max} must be selected to let the motor meet the expected vehicle power requirements. K_v must be selected in conjunction with battery voltage and propeller diameter to achieve maximum efficiency. If values of R_m, I_0, and mass can be derived from K_v and I_{max}, then the motor

model can be drastically simplified. Such a relationship was derived and included as an option within the model by analyzing a database of commercially available motors.

To arrive at good data regressions, motors from only one manufacturer were examined. The resulting regressions therefore do not represent universal surrogates, but still succeed in providing realistic parameter estimates for the purposes of model order reduction.

The Cobra motor brand was selected for analysis, since a large number of its motors were displayed on the Innov8tiveDesigns website with consistently labeled values for K_v, $I_{max}, R_m, I_0,$ and mass [40]. Data for 70 motors was extracted using a webscraper and compiled into a table. Response surfaces of $R_m, I_0,$ and mass as functions of K_v and I_{max} were then developed. Using logarithmic transforms, relatively strong fits were obtained, as is documented in Table 3. The surrogates performed comparably on both training and validation test data, indicating that no overfitting took place. The mass regression, for its part, boasted a coefficient of determination of over 0.98. The R_m and I_0 fits were not as strong, but still provide realistic approximations. The regressions were developed for values of I_{max} ranging between 5-75A and values of K_v ranging between 70 and 2500 RPM/V.

Thus, regressions of Cobra brand motor parameter data provided constraint relationships that made $R_m, I_0,$ and mass functions of K_v and I_{max}. These regressions reduced the number of required input motor parameters from five to two, thereby granting a simplified modeling option without great sacrifices in accuracy.

Table 3: Coefficients of determination for Cobra data regressions

	Mass fit R^2	R_m fit R^2	I_0 fit R^2
Training Data (51 data points)	.9873	.9615	.9674
Validation Data (18 data points)	.9829	.9422	.9364

7.5 Electronic Speed Controller (ESC)

7.5.1 ESC Background

The speed of the motor and propeller must be throttled to accommodate the different power requirements over the course of a vehicle's mission. This is commonly accomplished through an electronic speed controller, or ESC. A basic understanding of the functional principles of both brushless and brushed ESCs may be gained by referring to [41], [42], and [43]. In simple terms, ESCs control motor speed by sending pulses of electrical energy through the motor coils.

An ESC is typically modeled using three parameters: resistance, mass, and maximum current rating. It has been observed that ESC efficiency decreases with throttle [42]. Such an effect would impact the performance of the propulsion system at cruise or loiter, when the vehicle is not fully throttled. Furthermore, it is an effect that is not usually considered in existing propulsion models, illustrating a gap in current modelling techniques. A few researches have proposed models to more accurately capture the inefficiencies of ESCs [44] [45]. However, determining the parameters for these models requires extensive testing of many ESCs. Little experimental data has been published for ESCs, making it difficult to construct scalable data-based models and integrate them within a larger propulsion system module.

7.5.2 ESC Model Implementation

Two modelling options for ESCs were implemented within the propulsion module. The first assumes the ESC operates at a constant efficiency at all operating conditions. Typically, ESCs will have a high overall efficiency (between 0.95 and 1.0) if operating at full throttle. Additional resistive losses can be modelled by assigning a nominal resistance.

A second modelling option implements an empirically derived throttle-dependent efficiency relation, taken from Gong et al. [45]. This relation gives efficiency as a function of current flow, applied voltage, and throttle setting, and has the following form:

$$\eta_{ESC} = \delta(a_1 E_{ESC} + a_0) + bI_{ESC} + c_1 E_{ESC} + c_0 \quad (20)$$

In this equation, η_{ESC} is the ESC efficiency, δ is the throttle setting between 0 and 1, E_{ESC} is the voltage applied to the ESC, and I_{ESC} is the current flowing through the ESC. a_1, a_0, b, c_1, and c_0 are model parameters. Gong provides estimates of these parameters for a specific ESC based on experimental data. It is unlikely that these parameters apply to many other ESCs, however. Future research could focus on estimating these parameters for various ESCs. Until then, this throttle-dependent efficiency model may rarely ever be used in practice.

Electrical resistances can vary with temperature. The exact relationship is dependent on a variety of other factors, most prominently the component's insulation. The ESC model includes a functional method that outputs a resistance value based on current flow and ambient temperature. This function is integrated within the larger propulsion system analysis logic (described in a later section). In its default form, it merely passes the nominal ESC resistance without regard for the temperature and current input. The function's existence is justified by the potential flexibility it offers to future users who may desire to model changes in ESC resistance at high temperatures or current flows.

7.5.3 Reduced Order ESC Model

Additional regressions are included within the ESC model to reduce its complexity if desired. For instance, the model uses previously developed data-derived regressions of mass as a function of maximum current rating, $I_{ESC_{max}}$. Regressions developed by Winslow

et al. for either brushless or brushed ESCs predict mass for $I_{ESC_{max}}$ in the range 0A to 60A [41]. A regression by Bershadsky predicts mass for $I_{ESC_{max}}$ in the range 60A to 100A [38]. Furthermore, the ESC model includes an approximate relationship between ESC resistance R_{ESC} and $I_{ESC_{max}}$ (see Eqn. (21)). This relationship was adapted from the analysis tool eCalc.

$$R_{ESC} = 0.1589 * \left(I_{ESC_{max}}\right)^{-0.905} \tag{21}$$

These regressions eliminated two of the required ESC model inputs without significant loss in modelling accuracy.

7.6 Wiring

Copper wires have slight electrical resistances that produce power losses. They also contribute weight to the vehicle, and they have a maximum current rating, or ampacity, that limits permissible current flow.

The implemented wire model can predict resistance and mass if these are not exactly known. It requires inputs of length L_w, diameter D_w, ampacity a_w, and some material properties. Default material properties are those of copper. The wire resistance is calculated using a physics-based relationship described in [46] and shown in Eqn. (22).

$$R_w = \frac{4\epsilon L_w}{\pi D_w^2} \tag{22}$$

ϵ is the wire resistivity. Resistivity for copper and some other materials increases linearly with wire temperature. This effect is modeled in Eqn. (23).

$$\epsilon = \epsilon_0\left(1 + \alpha(T_a - T_0 + k_w I^2)\right) \tag{23}$$

α is the resistivity temperature coefficient, T_a is the ambient temperature, while T_0 is the reference temperature at which the reference resistivity, ϵ_0 is measured. For copper,

one may select T_0=0 °C, α=0.00451 K^{-1} ϵ_0 =15.4*10^{-9} Ωm [46]. The term $k_w I^2$ models temperature increases due to electrical power dissipation, where I is the current flow. Including the term requires further knowledge of the thermal resistance of the wire (quantified by the parameter k_w), which is usually not available early in design. Hence for most practical usages, k_w may be set equal to zero.

Wire mass m_w is computed using the material density, length, and diameter, as shown in Eqn. (24).

$$m_w = \frac{\rho L_w D_w^2 \pi}{4} \tag{24}$$

The default density is that of copper, equal to 8.96 g/cm^3. This equation neglects the mass contributions of the wire's insulation. It moreover assumes the wire is a solid cylindrical piece of metal, rather than a group of small strands, as may be the case.

Remote controlled aircraft are typically outfitted with silicone-insulated copper wires due to their light weight and wide temperature rating [47]. Using recommended current ratings for this wire type from a commercial datasheet, an approximate relationship between ampacity (a_w) and diameter (D_w) was derived [48]. The fit is presented in Eqn. (25) and Figure 28. This relation can eliminate either wire diameter or wire ampacity as an input parameter.

$$a_w = 1367.7\, D_w^2 + 590.74\, D_w \tag{25}$$

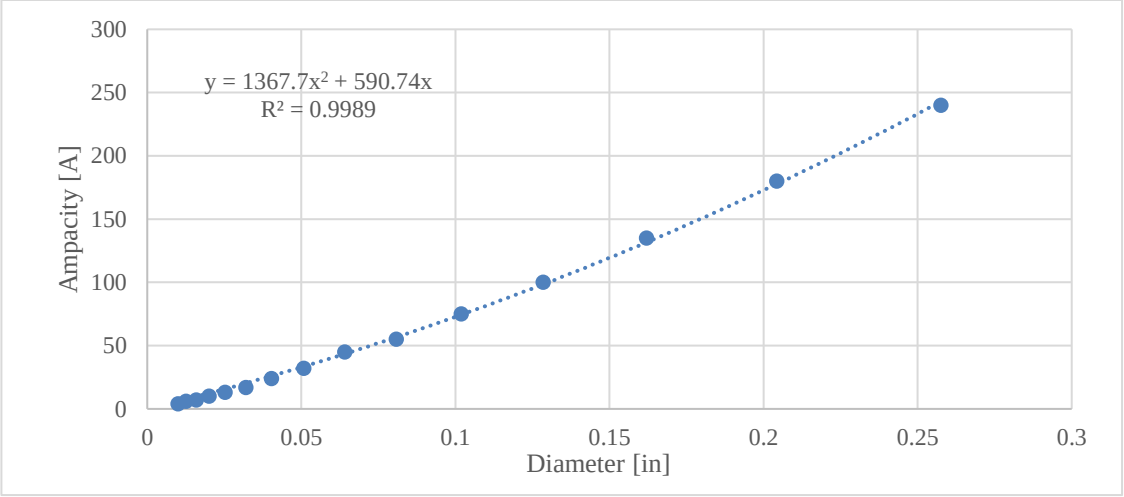

Figure 28: Ampacity recommendation vs. diameter for silicone insulated copper wires

In brief, the propulsion module incorporates a wire model that provides estimates of wire mass and resistance. It thereby offers higher fidelity modeling options for electrical wiring than are usually considered within propulsion modeling tools, all without contributing significant programming overhead.

7.7 Battery

7.7.1 Battery Background

The battery pack that supplies the electrical power. Electric vehicles tend to use rechargeable batteries, as these are the more economically viable option. A battery cell has several characteristic parameters. One is its nominal voltage, which is tied to the underlying chemistry. A second is the cutoff voltage. To understand this parameter, one must appreciate that as the battery discharges, its voltage sinks from an initial voltage, which may be larger than the nominal voltage. Allowing the voltage to sink lower than the stated cutoff voltage leads to permanent damage in the cell. A third parameter is the rated capacity of the cell, which is a measure of the amount of stored energy. Finally, the C-rating describes the cell's maximum advised discharge rate [49]. Additional battery terminology is summarized in [50].

Sadly, batteries do not supply a constant output of power throughout their discharge cycle. As the battery charge depletes, the voltage of the cell begins to sag in a nonlinear way [51]. Moreover, the sag increases as more current is drawn due to an internal battery resistance. Increasing current draw will also tend to reduce the effective capacity of the battery in a phenomenon known as the Peukert effect [27]. Decreasing the cell temperature or increasing the cell age will cause similar voltage sags and capacity reductions [52]. These general trends are illustrated in Figure 29.

In short, the ability of the battery to deliver power depends on where the vehicle is in the mission profile (i.e. on how much capacity has already been used) and on the magnitude of the power requirements of the mission segment (i.e. on how much current needs to go through the circuit). These effects are commonly neglected in initial sizing studies, but are extremely important to consider in performance analysis of electric aircraft. A battery discharge model that predicts available voltage of a battery cell as a function of state of charge, current draw, temperature, and age is therefore an essential part of a higher fidelity propulsion analysis code. Neither MotoCalc nor eCalc include this feature.

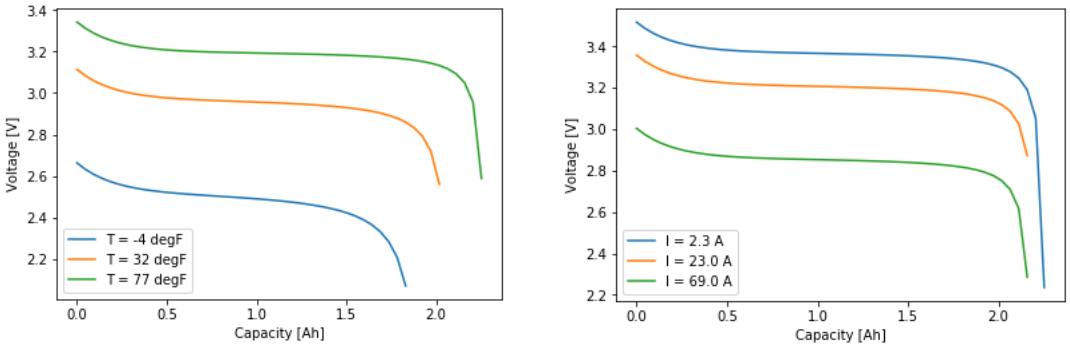

Figure 29: Effects of current and cell temperature on battery discharge behavior

7.7.2 Battery Model Implementation

The implemented battery discharge model was adapted from the generic battery model employed by the software Simulink [53]. The model's main purpose is to estimate cell

voltage V_c as a function of cell current I_c, capacity used q, and ambient temperature T_a (see Eqn. (26). Cell age was excluded as a simulation parameter, because its effect on battery discharge is hysteretic. The mathematical formulation is an adaptation of an experimentally validated model developed by Tremblay and Dessaint [51], and is modified to include the Peukert effect. A few additional simplifications were introduced, mostly to eliminate transient effects. Some options for reduced order modeling were also added. Section A.4 in the appendix contains a more detailed walkthrough of the implemented modeling equations.

$$V_c = f(q, T_a I_c) \tag{26}$$

It is common to arrange several battery cells into a larger battery pack. In so doing, cells will be wired in either series or parallel. The pack voltage and current draw (V_{pack} and I_{pack}) are determined from Eqn. (27) and Eqn. (28). Here, n_{series} and $n_{parallel}$ are the number of cells in series and parallel respectively.

$$V_{pack} = n_{series} V_c \tag{27}$$

$$I_{pack} = n_{parallel} I_c \tag{28}$$

Battery mass may be input from the outset if it is known. If not, an internally codified expression evaluates a mass estimate. The simplest way to model mass is to use a value of specific energy e_{sp} typical for the cell's chemistry. The mass of a battery cell m_c is then estimated using Eqn. (29), where Q_0 is the nominal cell capacity and E_0 is the nominal cell voltage. More accurate data regressions for battery mass may be included in the future.

$$m_c = \frac{Q_0 V_{nom}}{e_{sp}} \tag{29}$$

7.8 Integrated Circuit

The final aspect of the propulsion system module is the integrated circuit itself. The implemented circuit model is shown schematically in Figure 30. It was adapted from a similar model employed by Bershadsky [38]. At a given time, the current I_{pack} flows from the battery pack and then splits off. One branch, I_{sec}, supplies non-propulsive power consumers (e.g. servos, payloads, and avionics). Another branch, I_{ESC}, supplies one of potentially multiple parallel propeller-gearbox-ESC-wire combinations. Each parallel system is assumed to consist of the same types of components, operate at the same conditions, and receive the same current supply.

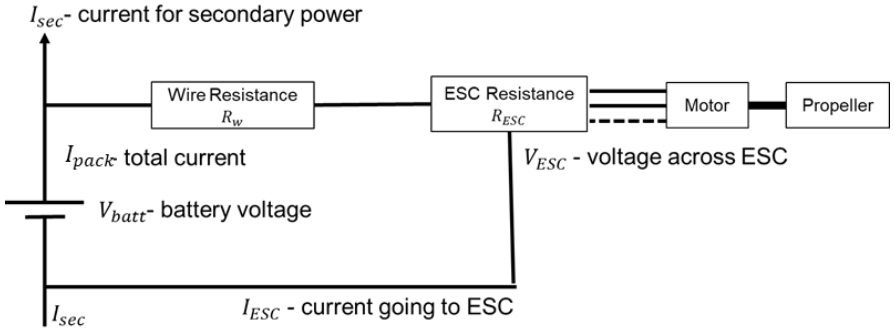

Figure 30: Circuit model used within the propulsion module

By applying Kirchoff's junction rule to the first node after the battery, one obtains Eqn. (30), where n_{drives} is the number of drive systems (wire, ESC, motor, gearbox, and propeller) used in parallel on the vehicle.

$$I_{pack} = I_{sec} + n_{drives} * I_{ESC} \tag{30}$$

Through a propeller-gearbox-motor analysis, it is possible to estimate the motor input power $P_{motor_{el}} = \Omega_m Q_m / \eta_m$. at an operating condition. The electrical power supplied to the ESC is given in Eqn. (31).

$$V_{ESC} I_{ESC} = \frac{P_{motor,el.}}{\eta_{ESC}} \tag{31}$$

Previous research indicated that ESC efficiency losses were generally a function of current, input voltage, and throttle setting:

$$\eta_{ESC} = f(I_{ESC},\, V_{ESC},\, \delta)$$

Comparatively little research addresses part-throttle electric propulsion modeling for UAVs. As a result, there is no clearly accepted definition of what throttle setting is. One approximation employed in eCalc and [38] is to define throttle as the ratio of supply voltage to the equivalent motor DC voltage $V_m = \Omega_m Q_m / I_m \eta_m$ determined through a motor efficiency analysis. This definition was used in the model and is shown in Eqn. (32).

$$\delta = \frac{V_m}{V_{ESC}} \tag{32}$$

Finally, the application of Kirchoff's loop rule yields Eqn. (33).

$$V_{ESC} = V_{pack} - I_{ESC}\,(R_w + R_{ESC}) \tag{33}$$

As stated previously, R_w and R_{ESC} are considered functions of the current I_{ESC} and the ambient temperature, T_a, though this dependency may often be ignored.

Using these equations, it is possible to converge on a value of I_{ESC}, V_{ESC}, I_{pack}, V_{pack}, and δ in the following manner:

1. Guess a value of I_{ESC}. A decent initial estimate is $I_{ESC} = P_{motor_{el}} / V_{pack}$, where V_{pack} is calculated for $I_{pack} = I_{sec}$.
2. Calculate R_w and R_{ESC} using the wire and ESC resistance model equations (inputs are the assumed I_{ESC} and a presumably known T_a)
3. Calculate I_{pack} using Eqn. (30)
4. Calculate V_{pack} using the battery model. It is assumed that the capacity used and ambient temperature are known.

5. Calculate V_{ESC} from Kirchoff's loop rule (Eqn. (33))
6. Calculate the throttle setting using the definition of throttle (Eqn. (32)). It is assumed that the motor operating condition is already known from a prescribed propeller operating state.
7. Calculate the efficiency of the ESC using the ESC efficiency model.
8. Recompute I_{ESC} by rearranging Eqn. (31) and compare it to the guessed value.
9. Return to step 2 unless I_{ESC} has converged.

This approach delivers the propulsion circuit state for a given propeller operating condition, battery capacity usage, ambient temperature, and auxiliary current draw. The propeller operating state is typically prescribed by a thrust value, as would be known during loiter or cruise, for instance. The dataflow for the thrust-prescribed analysis is illustrated in Figure 31. In other cases, the propeller thrust may not be known. Instead, the operating state may be defined in terms of a throttle setting, as in for instance a climb or acceleration segment. In this situation, a numerical solver varies the thrust input to the thrust-prescribed analysis until the desired throttle setting is achieved.

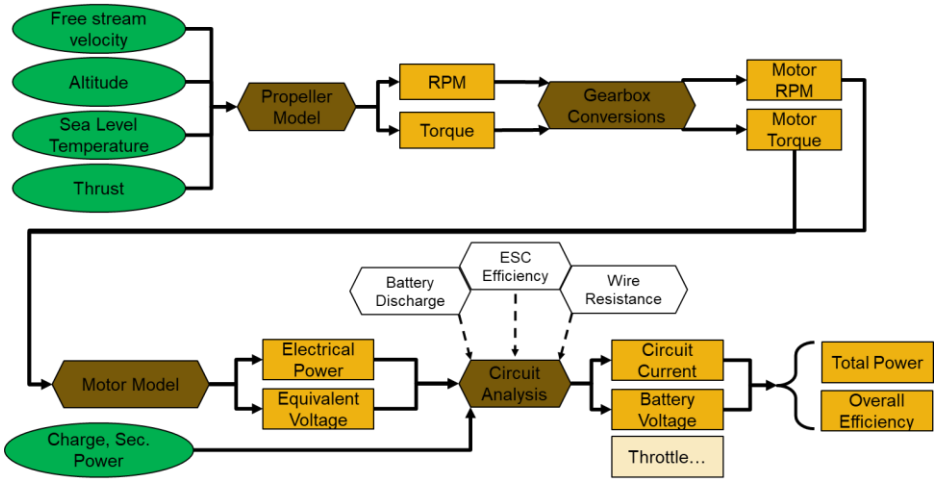

Figure 31: Dataflow of throttle-prescribed propulsion system performance analysis
(Green circles are inputs, gold boxes are outputs, and brown hexagons are models)

7.9 Validation

The previous sections described the development and implementation of an original electric propulsion model code for small fixed wing vehicle applications. Such a novel code needed to undergo a verification and validation process to confirm the its reliability and utility. A validation study was conducted to compare code predictions against the existing tool MotoCalc. The predictions were also compared to experimental motor-propeller performance data.

7.9.1 Procedure

The validation study was conducted in the following manner: Motor-propeller performance data was gathered for roughly 70 Cobra brand motors from the Innov8tiveDesigns website [40]. This data consisted of measurements of static thrust, current, and RPM for motor-propeller combinations at constant input voltages and full throttle. Roughly 2400 datapoints were collected. Each case included values of propeller pitch and diameter, input voltage, and motor parameters. From these inputs and assumptions of wire resistance, ESC efficiency, and ambient conditions, it was possible to model each case in the propulsion system performance code and execute a static full-throttle analysis. These same inputs were entered into MotoCalc to provide its prediction. The MotoCalc analysis was automated using the Python module pywinauto [54]. While the GUI automation did eliminate the need for manual input of each case, every button click or field edit lasted five seconds, making the automation infeasible for inclusion within an MDA environment. After compiling all the input data and results, the predictions were compared against each other and the experimental data.

7.9.2 Observations

The results of the study are illustrated in Figure 32. The plots show the predicted rotational rate, thrust, and current against the actual measured value. They also include the relative prediction residual $\left(\frac{Predicted-Actual}{Actual}\right)$ plotted against the measured value. The mean and standard deviation of these residuals are documented in Table 4.

The two analysis codes yielded similar predictions for propeller rotational speed. They were both consistently off by roughly 15% with a standard deviation in the error of 5-6.5%. For the thrust prediction, MotoCalc outperformed the novel code, overpredicting the actual value by only 20%, rather than 30%. The spread in residual was also narrower for MotoCalc. The new code made up for this deficit with improved predictions of the current flow. Overall, the new code's error spread in all three predictions was slightly larger than that of MotoCalc, but by a relatively small amount each time.

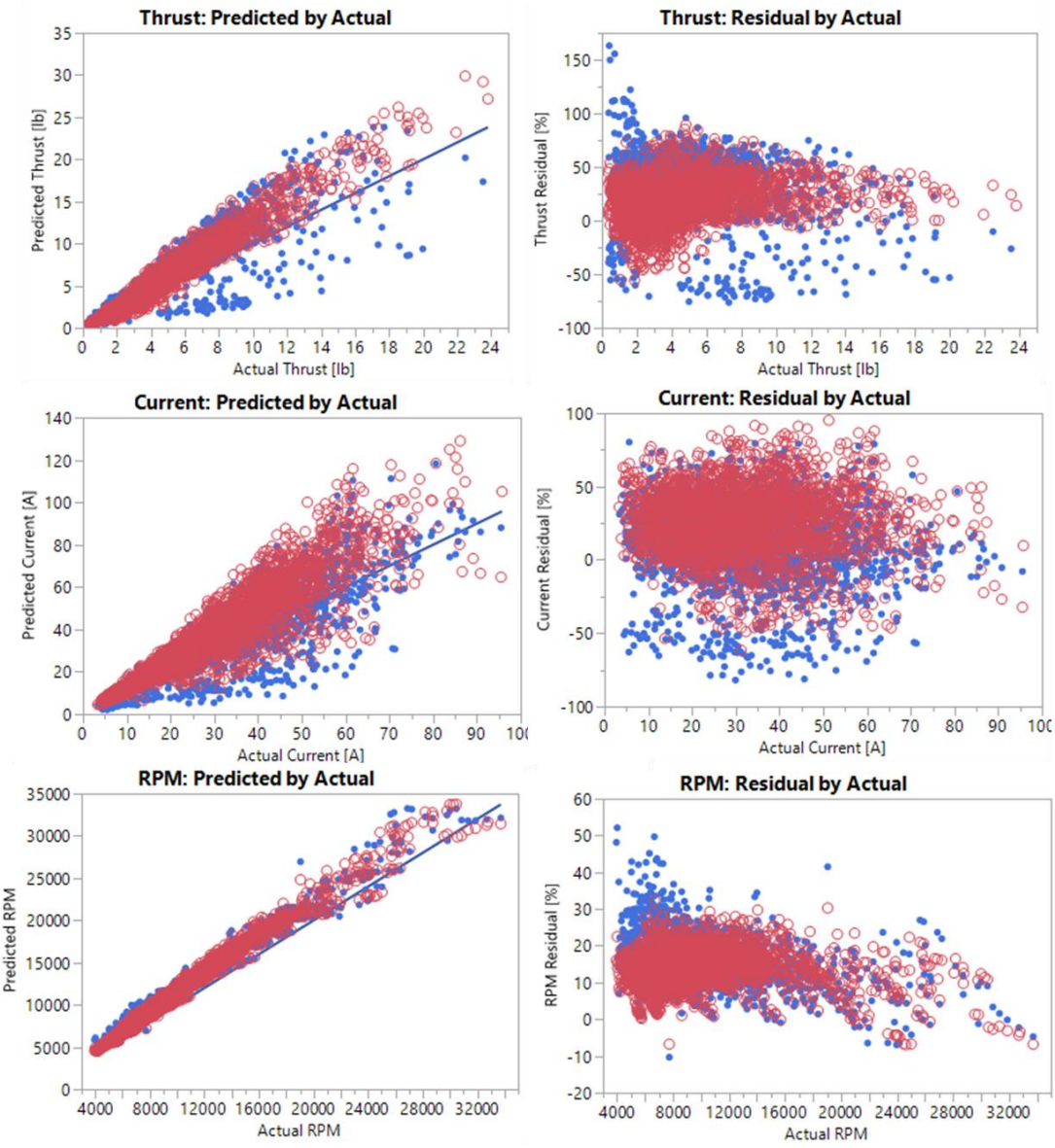

Figure 32: Verification study result plots

(Blue points correspond to predictions made by the newly developed analysis code; Red circles correspond to MotoCalc predictions)

Table 4: Residual means and standard deviations

	Average of Residual (%)			**Std. Deviation of Residual (%)**		
	RPM	*Thrust*	*Current*	*RPM*	*Thrust*	*Current*
Developed Model	15.56	31.51	12.40	6.516	27.38	26.59
MotoCalc	14.24	18.23	26.20	5.091	20.53	23.76

7.9.3 Conclusions

This study demonstrated that the implemented tool yields predictions comparable in magnitude and spread to those of a commercial tool, MotoCalc. Both tools tended to overpredict thrust, rotational speed, and current compared to actual experimental measurements. MotoCalc offered more accurate thrust predictions, but less accurate current predictions.

The differences in predictions between the developed tool and MotoCalc were traced to the propeller model formulation. This was proven in a separate study that replaced the parametric Xrotor-based propeller model with a surrogate of MotoCalc's propeller model. This surrogate was obtained from dynamic performance predictions gathered by the MotoCalc GUI automation. After incorporating this MotoCalc propeller surrogate within the newly developed code, the predictions were almost exactly equal to MotoCalc's own. This result indicated that the primary difference between the two codes lies in the propeller model formulation.

7.10 Propulsion Module Summary

In brief, this chapter documented the development, implementation, and validation of an original electric aircraft propulsion analysis code. This code was necessary to enable automated propulsion power loss analyses and component weight estimation for use in a multidisciplinary design analysis framework for unmanned electric flying wings. Its major features are a parametric propeller analysis employing the program Xrotor, a battery discharge model capturing the effects of current draw, state of charge, and temperature, a wire resistance and mass model, a reduced order motor model based on regressions of Cobra brand motor parameters, and optional component mass regressions.

A validation study demonstrated that the tool's predictive power was like that of commercially available software, with the main source of discrepancy lying in the propeller model. Moreover, the tool improves on existing software by enabling automated analyses, higher fidelity discharge modeling, and rubberized mass estimation. The code executes on the order of less than a second, thereby fulfilling the speed requirement for an MDA tool. Furthermore, it is open-source and can accommodate improvements or alterations by future users wishing to pursue their own niche design analyses.

It has therefore been demonstrated that the module captures the most significant interactions between electric propulsion, vehicle performance, and vehicle weight while satisfying the criteria of a tool suitable for MDA. Thus, the tool and the assumption underlying its development successfully address research question 2.2. Overall, the code constitutes a valuable contribution to the field of electric fixed wing UAV design.

The model's fidelity and power may be improved in the future through several means. These are enumerated below:

- Modeling of battery heating and thermal management
- Modeling of battery age effects on discharge behavior
- Modeling of aerodynamic impacts of ducts and propellers
- Improved estimates of blade aerodynamics for the parametric Xrotor analysis
- Improved ESC efficiency modeling capabilities
- Parameter estimation for nonlinear terms in the wire and motor models

CHAPTER 8. AERO-STABILITY

From a design standpoint, the aerodynamics discipline focuses on estimating the forces and moments that develop on a vehicle in different flow conditions. Knowledge of these quantities is indispensable for predicting vehicle performance and conducting structural analysis. Moreover, the aerodynamic characteristics of flying wings are inextricably linked with the vehicle geometry and volume.

Related to aerodynamics is the discipline of stability and control. Here the primary concern is to characterize the static and dynamic behavior of the vehicle and manipulate design variables to ensure the behavior is acceptable. To conduct a stability analysis, one requires adequate information about the aerodynamic forces and mass distribution of the aircraft. Designing for stability is more difficult for flying wings, as parameters that affect stability simultaneously affect vehicle aerodynamics. This constraint is absent in conventional aircraft and is the reason why aerodynamics and stability of a flying wing should be considered in conjunction.

These basic interactions between stability, aerodynamics, weight distribution, and performance demonstrated the need to include an aerodynamics and stability module within a multidisciplinary design analysis framework for unmanned electric flying wings. This realization led to the following research question:

Research Question 2.3
What modeling techniques will quantify the aerodynamic and stability behavior of unmanned electric flying wings while also satisfying the criteria (speed, fidelity, and automatability) of a tool suitable for MDA?

There are various methods of estimating aerodynamics, each with a different level of fidelity. These methods include historical data regression, numerical potential flow methods, analytical and empirically derived viscous drag models, and computational fluid dynamics (CFD). CFD analyses, while the most accurate, are slow and expensive, and often require a commercially licensed software. Historical regressions, by contrast, are fast, but offer an inadequate level of fidelity. This leaves potential flow methods and viscous drag models as the remaining analysis options that balance speed and fidelity. This process of elimination suggested the following assumption to address Research Question 2.3:

Assumption 2.3

If potential flow methods and viscous drag models are used in conjunction with weight distribution estimates, then the aerodynamic and stability characteristics of flying wings can be quantified within an MDA framework.

The remainder of the chapter documents the implementation and evaluation of such an aero-stability module.

8.1 All-Wing Aero-Stability Considerations

The first step in developing an aero-stability module for flying wings was to better understand the interactions between aerodynamics and stability on this configuration. This helped identify what capabilities and fidelity levels the modeling tools needed to provide.

The most prominent requirement that stability imposes on the aerodynamics of flying wings relates to longitudinal or pitch stability. As mentioned in the introduction and proven in section A.5 in the appendix, pitch stability occurs under two conditions. First, the aircraft must have a positive static margin. Second, the moment about the neutral point of the wing must be positive. This latter requirement is unique to the flying wing configuration and

attempting to meet it leads to consequential tradeoffs in aerodynamic efficiency. These were mentioned previously, but will now be explained in greater depth.

8.1.1 Reflex Airfoils

The first approach to ensuring a positive moment coefficient about the neutral point is to use an airfoil with a positive moment coefficient. Unfortunately, traditional low-speed, positively cambered airfoils that offer high lift for low drag penalties all possess negative and sometimes strongly negative moment coefficients. Consequently, the most efficient kinds of low-speed airfoils are unsuitable for all-wing aircraft. Airfoils with negative camber would give a positive moment coefficient, but at considerable cost in aerodynamic efficiency.

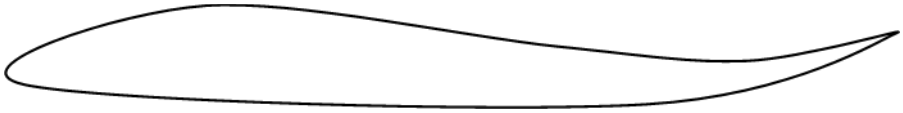

Figure 33: Generic reflex airfoil

The apparent best compromise solution is to employ reflex airfoils. Reflex airfoils have a characteristic s-shaped profile, as shown in Figure 33. The front part looks much like a cambered airfoil, and it offers many of the advantages of this geometry. Towards the rear, the camber inflects to give the airfoil an upturned tail. At this tail, the pressure distribution crosses over, as depicted in Figure 34: The pressure on the top of the wing becomes high, while that on the bottom becomes low. This results in a downward force. Naturally, such a force counteracts lift and degrades aerodynamic efficiency. Yet it also provides a positive moment about the aerodynamic center. The resulting inefficiency is less than that of a negatively cambered airfoil, making reflex airfoils the most aerodynamically efficient known positive-moment airfoils.

This discussion reveals that the aerodynamic characteristics of the airfoil used on a flying wing are of immense importance in determining the stability behavior of the aircraft. The modeling tools employed in an aero-stability module should therefore be capable of predicting and considering the lift, drag and moment coefficients of the wing airfoils.

To find additional information on the requirements of flying wing airfoils, as well as lists of airfoils suitable for all-wing vehicles, one may consult [6], [55], [56], and [57].

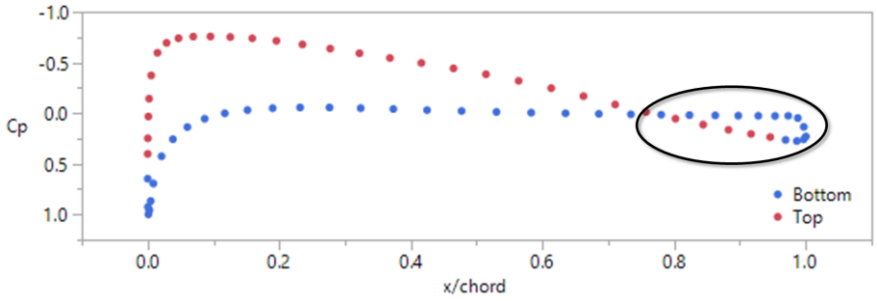

Figure 34: Pressure coefficient plot for a generic reflex airfoil

8.1.2 Lift Distribution of Sweptback Wings

The second design option that yields the desired positive moment coefficient involves a sweptback, washed-out wing planform (recall Figure 6). By sweeping the wing back and reducing the lift on the tips through geometric twist, the wing will generate more lift in the front by the root than in the back by the tips (see Figure 35). The result is a positive moment coefficient about the neutral point, even if the moment coefficient of the airfoils is not itself positive. To avoid strong twist, it is still necessary for the airfoil moment coefficient be close to or larger than zero. Symmetric or reflex airfoils are most suitable here.

Clearly, such a lift distribution is inefficient, since the tips are not generating as much lift as they could. Indeed, the wingtips may sometimes need to produce negative lift depending on the location of the center of gravity. The reason for this dependency is that the necessary trimming pitch moment increases as the moment arm of the center of gravity

increases, i.e. as the static margin grows. To achieve this trimming moment on a sweptback flying wing, the load on the wing tips is reduced and the load on the wing root is increased, as illustrated in Figure 35. This relation between lift distribution and center of gravity location is a unique feature of the flying wing configuration that further reinforces the need to treat stability and aerodynamic analysis in conjunction. Furthermore, it becomes evident that any practical aero-stability module for flying wings must model the effects of sweep, twist, and control surface deflections.

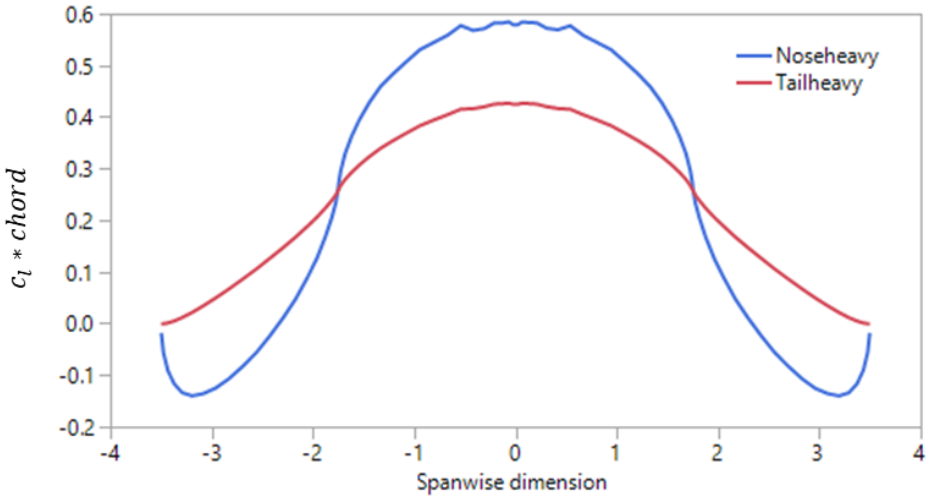

Figure 35: Lift distribution at trim for a generic sweptback flying wing

8.1.3 Center of Gravity Effects

As mentioned in the previous paragraph, flying wings are characterized by special constraints on the positioning of the center of gravity. Longitudinal stability requires that the center of gravity lies ahead of the neutral point, as illustrated in Figure 36. According to empirical observations by the Horten Brothers and Nickel and Wohlfahrt, this necessary condition may not always be sufficient [6]. These flying wing designers state that for good handling in turbulent conditions, the center of gravity must also be ahead of the "C-point", which is the position of the center of lift for a lift distribution that is strictly proportional to wing section chord [58]. Nickel and Wohlfahrt ultimately recommend a 6-12% static

margin with respect to the C-point. The designers provide no theoretical explanation for this requirement, making the conclusion somewhat suspect. It is unclear how they were able to draw a correlation between the unsatisfactory handling qualities and the C-point, which seems to have no relevance in any other contexts. Regardless, it is evident that aero-stability analysis must include a quantification of the static margin with respect to the neutral point and the C-point.

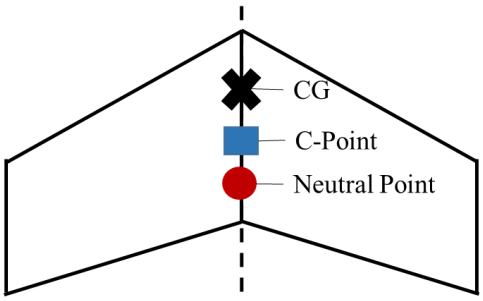

Figure 36: Important points on a flying wing

Stall behavior places additional constraints on the center of gravity's permissible range in the sweptback configuration [6]. Depending on the static margin, the vehicle may exhibit either nose-dive or rear-up stall.

A nose-dive stall is characterized by a stabilizing pitch-down moment (right image in Figure 37). This moment, though stabilizing, may lead to sudden crash landings. The nose-down moment arises because there will be a sudden loss in lift at the front of the wing, while the aft wing tips still produce lift. This occurs for noseheavy sweptback flying wings, where the front of the aircraft must produce more lift to achieve trim. The desire to avoid sudden nose-dives leads to an upper limit on permissible static margin.

By contrast, a rear-up stall is an unstable behavior, characterized by a sudden nose-up moment when stall occurs (left image in Figure 37). It occurs if the wingtips produce a

greater load and stall before the wing root. This would be the case for tail-heavy flying wings. Avoiding rear-up stall is an additional motive for achieving a positive static margin.

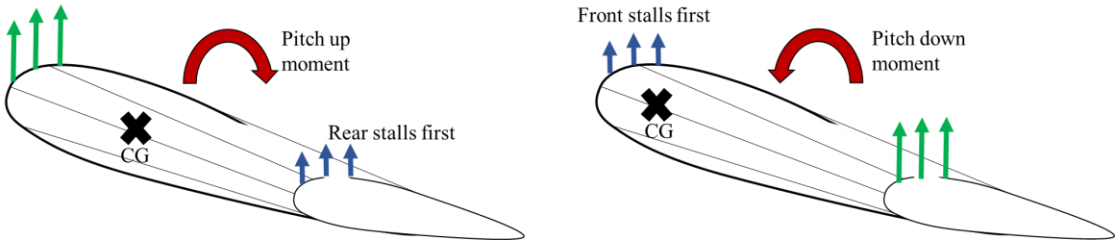

Figure 37: Flying wing stall phenomena – left: rear-up stall due to tail-heaviness – right: nose-dive stall due to nose-heaviness

This discussion of stall behavior reinforces the importance of static margin in estimating flying wing aerodynamics and stability. An aero-stability module for the all-wing configuration must include the mass and center of gravity as crucial modeling inputs, especially when estimating maximum C_L.

8.1.4 Module Requirements

From this review of flying wing aerodynamics and stability, as well as basic knowledge of aircraft design, it becomes possible to make a list of important outputs of an aero-stability module:

1. Airfoil aerodynamics: The module must be able to estimate the lift, drag, and moment of airfoils used on the wing.
2. Drag polar: The drag polar is useful in performance as it permits estimation of conditions for minimizing power or thrust and quantifies the aerodynamic efficiency of the aircraft.
3. Maximum lift coefficient: Knowledge of the maximum lift coefficient is critical for imposing limits on aircraft landing and turning performance.
4. Lift distribution: The lift distribution for different conditions is a critical input for structural analysis. Moreover, qualitative examination of the lift distribution gives

insight into the aerodynamic efficiency of the aircraft and its susceptibility to different types of stall.

5. Angle of attack, control surface deflections, and stability derivatives for trimmed flight: It is important to verify that these parameters lie within reasonable ranges. Such verification not only indicates stable designs, but also to provides a sanity check on the analysis.
6. Dynamic modes: Knowledge of these modes is critical for evaluating the aircraft's handling qualities.

8.2 Existing Aero-Stability Modeling Tools

Fortunately, there already exist various computer programs for aerodynamic and stability analysis of aircraft. The module as implemented incorporates the capabilities of three open-source programs: OpenVSP, AVL, and Xfoil (see Figure 38).

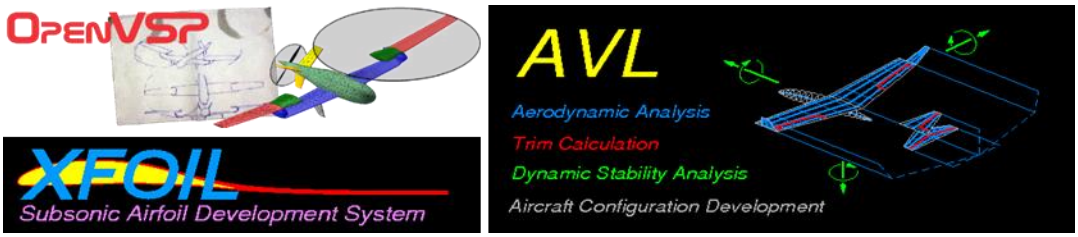

Figure 38: Logos for support programs used within the aero-stability module

8.2.1 Athena Vortex Lattice

Athena Vortex Lattice (AVL) is an openly licensed vortex lattice code developed at MIT by Mark Drela and Harold Youngren [59]. The software is a potential flow solver, and its domain of validity is restricted to larger aspect ratio wings at low angles of attack and low Mach numbers. Given a geometry and a mass distribution, the software can solve for the lift distribution, induced drag, and dynamic modes at trim conditions, taking deflections of control surfaces into account. At these trim conditions, AVL computes static

stability derivatives and dynamic modes of the aircraft. Like most Drela-Youngren tools, AVL can be easily automated with the subprocess module in Python.

8.2.2 Xfoil

Xfoil is a widely used 2D flow solver developed by Mark Drela and Harold Youngren [34]. The software predicts the pressure and skin friction distribution around an airfoil given a geometry, Reynold's number, Mach number, and angle of attack. Within the aero-stability module, Xfoil is primarily used to estimate the maximum lift coefficients of airfoils in support of a $C_{L_{max}}$ calculation.

8.2.3 OpenVSP

OpenVSP is a parametric aircraft geometry modeling software developed by NASA engineers [26]. It includes a parasite drag analysis that codifies many viscous drag models.

8.3 Aero-Stability Module Implementation

The aero-stability module consists of automation wrappers around the three external programs: Xfoil, AVL, and OpenVSP. Xfoil produces drag polars of arbitrary airfoils. AVL provides the main analysis routine, which is to evaluate the force distribution of the wing at different operating conditions and compute the stability derivatives and dynamic modes. OpenVSP calculates a zero-lift drag coefficient. This computation is necessary, as vortex lattice methods are based on potential flow theory, which do not capture viscous effects. The impact of viscosity is therefore modelled as a constant parasitic drag coefficient contribution. In this section, the capabilities and functionality of each of these wrappers will be discussed in turn.

8.3.1 Xfoil Wrapper

The Xfoil wrapper is a python programming class that accepts the inputs:

- Airfoil coordinate file name
- Airfoil folder path
- Xfoil executable file path
- e^n transition parameter, N_{crit}
- Airfoil thickness scaling factor
- Xfoil iteration limit

The wrapper supports several functionalities. First, it can generate a plot of the airfoil and output its x-y-coordinates. This is simply achieved by parsing the input airfoil file provided in Selig .dat format [60]. Second, the wrapper can execute an angle-of-attack sweep on the airfoil. The inputs to this method are a range of angles-of-attack and a Reynold's number. The wrapper issues the commands to Xfoil through the functionality in the Python subprocess module [61]. As a result, Xfoil outputs a file with the lift, drag, and moment coefficients of the airfoil at each angle of attack. This file is parsed and output as a drag polar programming class.

The drag polar class can be instantiated either with a path to an Xfoil output file or by inputting lift and drag data directly. In addition to storing the drag polar data, the class automatically processes the data, providing a quadratic drag polar fit ($C_D = C_{D0} + K_1 C_L + K_2 C_L^2$), the maximum lift coefficient, and a linear fit to the lift curve ($C_L = m * \alpha + b$) for data points in the angle of attack range between [-2, 8] degrees. The linear domain of the lift curve was hardcoded in this manner, as attempts at developing a method to detect it automatically proved fruitless. The class has additional methods for plotting data and obtaining the lift and drag coefficients for either maximum C_L/C_D or maximum $C_L^{1.5}/C_D$. Figure 39 shows some example plots that are readily produced by this class. Overall, the Xfoil wrapper constitutes a useful utility in its own right. Its range of functionality may be expanded in the future.

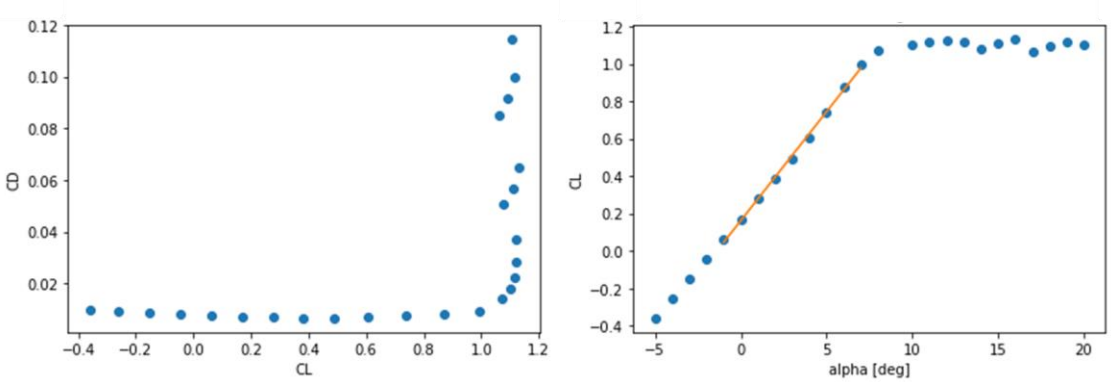

Figure 39: Airfoil data plots generated by the drag polar class – lift curve includes line showing linear fit estimate

8.3.2 OpenVSP Wrapper

The OpenVSP automation wrapper computes the parasitic drag coefficient in an automated fashion given a wing geometry class, a Reynold's number, and a Mach number. The similarity parameters are derived from inputs of flow speed, density, and temperature. The wrapper permits user control over the drag modelling through the choice of turbulent drag model (e.g. Blasius power law), form factor equation (e.g. DATCOM), percent laminar flow, and a constant $C_{D,0}$ addition factor (to account for additional drag sourcesnot included in the geometric model).

The aero-stability module interacts with OpenVSP through its scripting capabilities. The Python wrapper generates a text file that contains commands to generate the wing geometry and perform the parasite drag analysis. The wrapper then opens OpenVSP and runs the script, which produces result files that the wrapper parses.

The OpenVSP parasite drag analysis can last up to a second. Considering how often a new parasite drag estimate may be required during performance analysis, this execution time verges on unacceptably slow. Consequently, the wrapper includes the option to create a surrogate of the parasite drag calculation. This involves executing the analysis for five values of Reynold's number. The results are then fit to a power law of the form seen in

Eqn. (34). Thereafter, the wrapper ceases to employ OpenVSP and instead uses the power law to predict parasite drag rapidly. A plot comparing this drag surrogate to OpenVSP predictions is included in Figure 40 to demonstrate the validity of this fit model.

$$C_{D,0} = a * (Re)^b \tag{34}$$

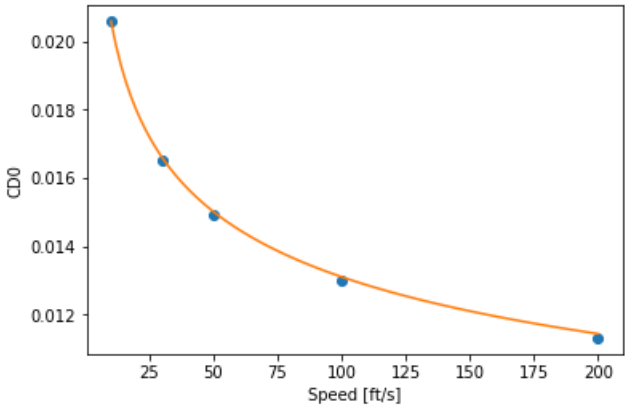

Figure 40: $C_{D,0}$ results from an automated parasite drag analysis of a generic flying wing
(points are direct OpenVSP results – the curve is the power law regression)

8.3.3 Athena Vortex Lattice Wrapper

The third and most important component of the aero-stability module is the automation wrapper around AVL. The wrapper automatically generates an AVL input file based on an input geometry class. It can then perform two distinct types of analyses:

The first is a flow solution at a fixed angle of attack or a fixed value of C_L. This analysis only requires knowledge of the operating condition and wing geometry. This analysis is primarily used to quickly obtain lift and drag predictions for a range of angles of attack, from which one may develop a drag polar model. However, this analysis does not include the effect of weight distribution on lift distribution in trim, which for flying wings may be considerable.

The second, more valuable analysis is a steady trim flow solution. Two trim situations may be modeled: The first is steady level banked flight. The second is steady climbing

unbanked flight. Future improvements could add additional trim situations to model. The routine requires inputs of weight and weight distribution. The user must also specify which control surfaces are responsible for pitch and roll balance. The module supports flaperons by modeling an aileron and a flap at the same position [62]. As implemented, the yaw moment can only be balanced out by varying the sideslip angle of the vehicle. In future, the option to employ additional control surfaces for yaw balance may be incorporated. AVL ultimately solves for the aerodynamic coefficients, angle of attack, control surface deflections, stability derivatives, spanwise force distributions, and dynamic modes at the trim position. It must be noted that the impact of thrust on the aircraft's pitching moment cannot be modeled within AVL. The results are output to files, which the wrapper parses and organizes. The data obtained from the analysis can be readily examined and visualized, as exemplified in Figure 41. In future, the wrapper may be enhanced with added capabilities, such as obtaining hinge moments, conducting a looping flight analysis, or saving animations of the vehicle's modal behavior.

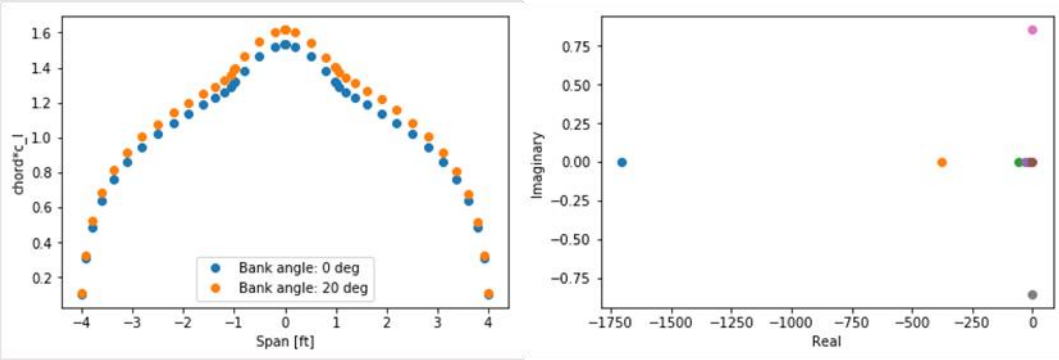

Figure 41: Data plots for trim analysis results
(left: $c * c_l$ distribution at two bank angles; right: dynamic modes in the complex plane)

8.3.4 Integrated Aero-Stability Module

The wrappers for AVL, OpenVSP, and Xfoil are integrated within a single aero-stability programming class that connects shared analysis inputs and outputs. The

modelling inputs to the class include the vehicle geometry class, vehicle mass properties (mass, center of gravity, and moment of inertia tensor), and any settings specific to each of the automation wrappers. The class then provides methods to accomplish the required modelling tasks:

Trim Analysis

The steady banked or climbing trim analysis executes the previously described trim analysis within AVL. The only modification to the method is the inclusion of a parasite drag estimate prior to executing AVL.

The outputs are divided into two camps. The first set is a table of sectional aerodynamic coefficients (c_l, c_d, c_m) at discrete locations along the span. The second set contains the aerodynamic coefficients, stability derivatives, dynamic modes, and neutral point X_{np}. Static margin with respect to the neutral point SM is easily computed from Eqn. (35). The length measure by which the margin is normalized is the mean aerodynamic chord MAC, computed within the geometry class using Eqn. (36) [63]. In this equation, b is the wingspan, c is the wing chord, and y is the spanwise direction. Figure 42 provides a graphical interpretation of these parameters. Additionally, a static margin with respect to the C-point SM_C is computed for the sake of completeness (Eqn. (37)). The chordwise position of the C-point X_C is calculated using Eqn. (38). This equation was adapted from [6], and assumes that the aerodynamic center of the wing airfoils x_{ac} is at the quarter chord.

$$SM = \frac{X_{np} - X_{cg}}{MAC} \tag{35}$$

$$MAC = \sqrt{\frac{1}{b}\int_{-b/2}^{b/2} c(y)^2 dy} \tag{36}$$

$$SM_C = \frac{X_C - X_{cg}}{MAC} \tag{37}$$

$$X_C = \frac{1}{S}\int_{-b/2}^{b/2} c(y)x_{ac}(y)dy \tag{38}$$

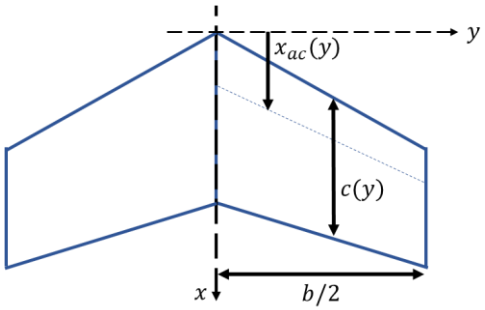

Figure 42: Illustration of parameters used in Eqns. (35) through (38)

$C_{L,max}$ and Stall Speed Estimation

The potential flow modeling formulation employed within the aero-stability module is incapable of directly predicting flow separation. Nevertheless, it is still possible to obtain an approximate upper bound on lift. The method for doing so requires knowledge of the maximum sectional lift coefficient of the wing airfoil [62]. If the vortex lattice solution predicts sectional lift coefficients in excess of the maximum 2D lift coefficient, then the solution can no longer be trusted. Hence, one may approximate the stall angle as the angle of attack at which any of the sectional lift coefficients along the span exceed the maximum airfoil lift coefficient. This general idea is illustrated in Figure 43.

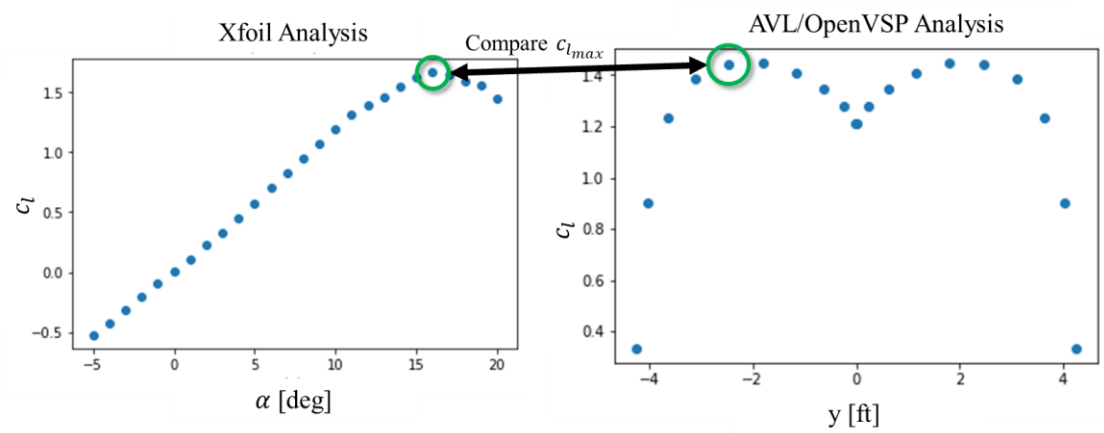

Figure 43: Illustration of method for estimating stall angle of flying wing
(right: lift curve of the airfoil employed on the wing at the flow Reynold's number; left: c_l distribution along the span of the wing)

The method is implemented in a simplified form within the aero-stability module. First, the maximum c_l of the root airfoil of each section is identified through an Xfoil analysis at an initial stall speed guess. $c_{l,max}$ is taken to be the minimum value found for all the root airfoils. If this analysis encounters an error, the program assumes a default $c_{l_{max}}$ of 1.0. The program then executes a steady trimmed AVL analysis at the same speed and identifies the maximum c_l along the span. This value is compared against $c_{l_{max}}$. The program uses a root-finding algorithm to identify the flight speed at which the two values become equal. This speed is output as the stall speed and the corresponding value of C_L is the maximum lift coefficient of the wing.

As a result, the analysis can capture the impact of center of gravity location on the maximum lifting capability and stall speed of a flying wing. This is demonstrated in Figure 44, which shows a study on a generic sweptback flying wing, in which $C_{L_{max}}$ is observed to decrease with static margin. This occurs because as the static margin increases, the necessary pitch up moment to achieve trim increases in magnitude. To attain trim, the root of the sweptback wing must then produce a higher load, i.e. the sectional values of c_l

increase, even while the airspeed remains unchanged (recall Figure 35). Consequently, the airspeed at which the maximum observed c_l on the span exceeds the maximum airfoil c_l is higher than it was for a smaller value of static margin. Correspondingly, the value of $C_{L_{max}}$ at which this stall condition occurs is lower.

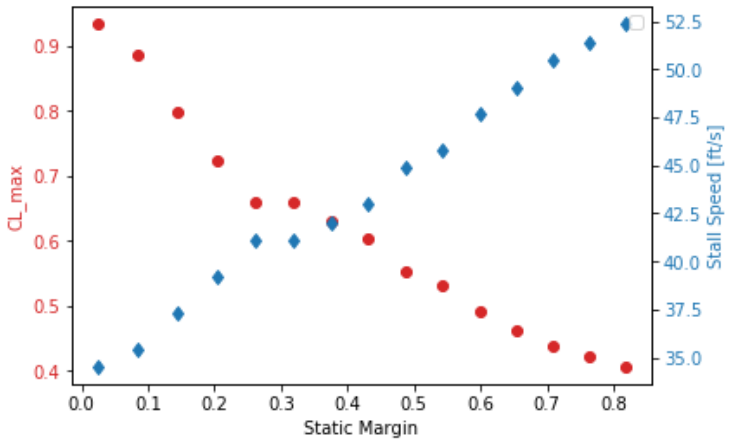

Figure 44: Dependence of $C_{L_{max}}$ and stall speed on the center of gravity location for a generic sweptback flying wing

If the trimmed flight analysis fails for some reason, the program attempts the same process using a stick-fixed analysis, this time varying angle of attack at a fixed speed. At this point, the effect of center of gravity no longer affects the analysis, so fidelity and realism are lost. Should this analysis in turn fail to converge, the C_L at a default guess of stall angle is output as $C_{L_{max}}$. The stall speed in these situations may be found through a convergence algorithm with the following steps:

1. Guess a value for stall speed.
2. Compute $C_{L_{max}}$ at that speed with the stick-fixed analysis (i.e. vary the angle of attack until the maximum sectional lift coefficient equals the 2D $c_{l_{max}}$).
3. Recalculate the stall speed using Eqn. (39).
4. Return to step 2 unless the stall speed has converged.

$$v_{stall} = \sqrt{\frac{W}{S}\frac{2}{\rho C_{L_{max}}(v_{stall})}} \tag{39}$$

The method as implemented loses accuracy if the wing has aerodynamic twist or strong taper. In such cases, the maximum 2D c_l of a wing slice could vary significantly along the span due to changes in geometry and Reynold's number. Such a variation is not considered when solving for stall speed, since the method does not consider where along the span the maximum c_l is found. Nevertheless, given that the method itself only offers a rough estimate, this breakdown in modeling rigor is deemed acceptable.

Analysis executions indicated that the stall speed and $C_{L_{max}}$ computation was often time intensive. Consequently, a capability was included to generate a polynomial fit model of $C_{L_{max}}$ for varying values of kinematic viscosity. This fit model can be generated with a simple command. Thereafter, the fit model is used to rapidly compute stall speed and maximum lift coefficient at any altitude. In that respect, the surrogate is like the power law surrogate that can optionally be used for parasite drag estimation.

Drag Polar

The aero-stability module includes a method that executes an analysis for a range of angles of attack. The aerodynamic coefficients from each analysis are assembled into a table and used to define a drag polar programming class, as used by the Xfoil wrapper. The quadratic drag polar model invariably provides an excellent fit to AVL's predictions, due to AVL's linear analysis. This vehicle drag polar obtained from this analysis may be used within performance modeling to compute drag estimates given prescribed values of lift in situations where a trim analysis either fails or is inappropriate.

As suspected, the drag polar varies significantly depending on whether the aircraft is in a trimmed state or not. For instance, Figure 45 shows how for a generic sweptback wing, the drag at a given C_L increases considerably if the vehicle is trimmed. Such a result provides further evidence of the need to include stability and trim considerations within the aerodynamic assessment of flying wings.

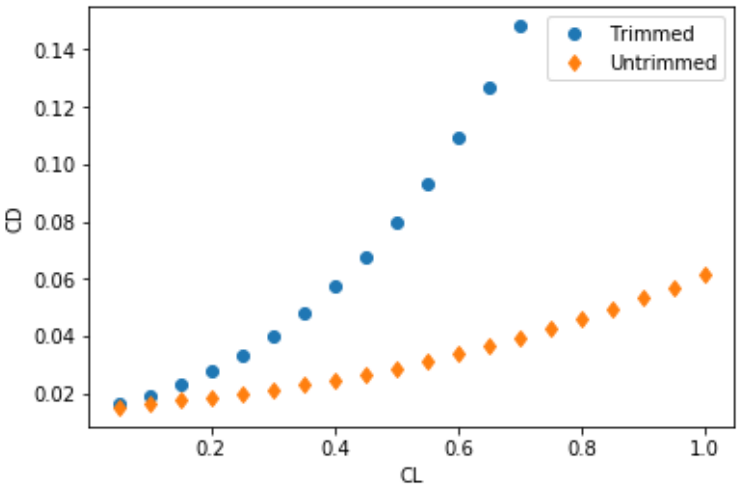

Figure 45: Stick-fixed vehicle drag polar

8.4 Aero-Stability Module Summary

This chapter addressed the development of an aero-stability module for an unmanned electric flying wing MDA framework. Such a module was essential for accurate estimation of performance, determination of handling qualities, and interface with structural analysis. Vortex lattice methods and viscous drag models were proposed as the modeling tools capable of providing the desired functionality at a sufficiently high fidelity level and computational speed. The module's capabilities included trim analysis, drag polar generation, maximum lift and stall speed computation, modal analysis, and lift distribution estimation. These functions were achieved by developing automation wrappers around the external programs AVL, Xfoil, and OpenVSP. Small case studies demonstrated that the module could perform the required modeling tasks in an automated fashion with great

rapidity and limited user effort. These demonstrations proved that the vortex lattice methods and viscous drag relations could model the most relevant aero-stability phenomena of flying wings in a manner suitable for incorporation within an MDA framework. In particular, it was possible to capture the interaction between weight distribution and lift distribution and stall speed, which is an essential interdisciplinary coupling for flying wings. These demonstrated capabilities indicate that Research Question 2.3 was successfully addressed.

CHAPTER 9. AIRCRAFT PERFORMANCE

9.1 Motivation

Performance estimation is a critical step within aircraft design. Many vehicle requirements such as maximum speed, range, endurance, takeoff field length, rate of climb, and service ceiling are performance metrics. Moreover, mission modeling, which is crucial for predicting to what degree a vehicle will successfully accomplish its intended mission, relies strongly on performance equations.

Performance is an inherently interdisciplinary field, synthesizing aerodynamics, propulsion, and weight distribution. It is therefore appropriate and indeed necessary to include performance estimation and mission modeling within an MDA framework. Including them will not merely permit prediction of design metrics for requirements verification, but will also demonstrate that the multidisciplinary framework can capture interdisciplinary interactions. This chapter discusses the implementation of an aircraft performance module used within the unmanned electric flying wing MDA framework.

9.2 Modifications for Electric Propulsion

In fundamental terms, performance estimation for unmanned electric flying wings differs little from that for conventional fuel burning aircraft. The general equations of motion derived using classical mechanics still apply. Some important distinctions do arise with the use of electric power: In contrast to fuel-burning aircraft, electric vehicles do not decrease in weight as energy is consumed. Available energy for electrically powered vehicles is most directly related to the state of charge of the battery. This charge is depleted as electrical current flows through the circuit. In theory, total onboard energy is exhausted

when the total available charge is depleted. But in practice, energy is exhausted when the battery voltage sinks below the cutoff voltage. This may occur even before the nominal charge of the battery has been totally used up. This physical phenomenon is captured by the battery discharge model and is essential for predicting power availability over the course of a mission.

9.3 Performance Module Implementation

The performance module was developed as a separate programming class for the MDA framework. It codifies commonly used performance equations. These equations are accessible as class methods that pass inputs to automated and interconnected aerodynamic and propulsion analyses.

9.3.1 Setup

As shown in Figure 46, the performance module must be linked to an aero-stability class and a propulsion class. Through these connections, the module gains access to methods for predicting lift, drag, thrust, current flow, and other quantities at different operating conditions.

Aerodynamic analyses may be conducted in two ways. The first is to employ AVL's trim analysis to obtain the aerodynamic states of the aircraft in a trimmed state. This is used for any situation in which one may assume the aircraft is level and quasi-steady. For other situations such as climbing flight, or in cases where the AVL analysis fails to converge on a flow solution, a quadratic drag polar function is employed to predict drag given a known value of lift. To reduce computational costs, the drag polar is estimated at a single reference condition. This polar is stored as a class attribute and updated only when the reference conditions or the aero-stability class are altered.

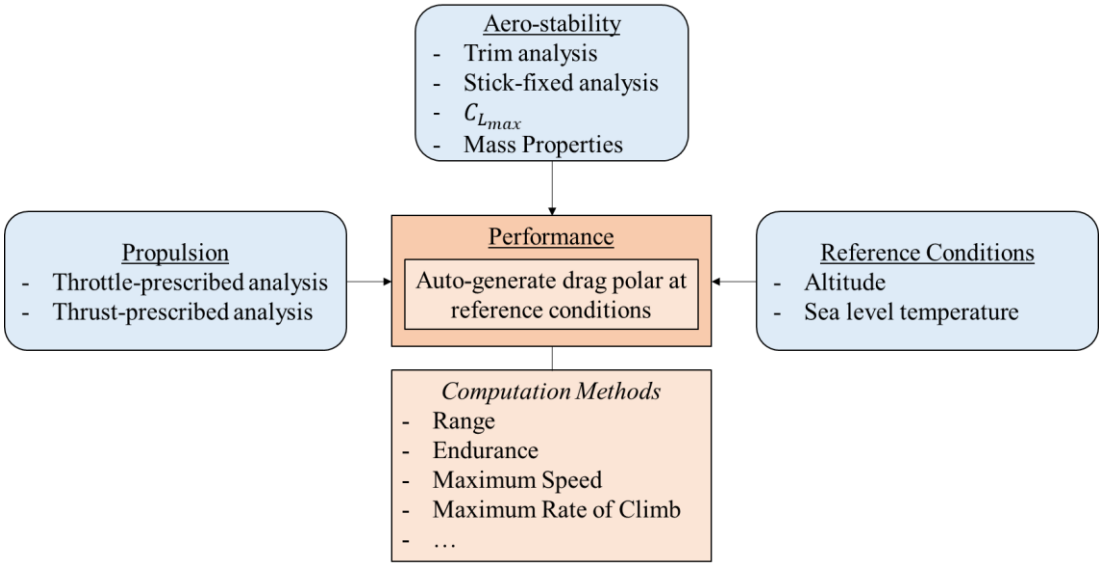

Figure 46: Support modules and input parameters for the performance module

9.3.2 Environmental Model

Atmospheric parameters such as density, temperature, or viscosity are important simulation parameters. The framework employs the Python module scikit-aero to compute values of ambient density ρ_a, pressure P_a, and temperature T_a at arbitrary altitudes and sea level temperatures using the COESA standard atmosphere model [64]. Offsets from standard sea level temperature are modeled by adding the offset to the predicted temperature, keeping pressure the same, and then recomputing density from the equation of state, shown in Eqn. (40). Here, R_a is the gas constant of air, equal to 1716 $\frac{ft\ lb}{slug\ R}$.

$$\rho_a = \frac{P_a}{R_a T_a} \tag{40}$$

Viscosity μ_a is modeled using Sutherland's formula as presented in [65] and repeated in Eqn. (41), where $T_0 = 518.7\ R$ and $\mu_0 = 3.62 * 10^{-7}\ \frac{lb\ s}{ft^2}$. A graphical depiction of the dataflow for determining atmospheric parameters is presented in Figure 47.

$$\mu_a = \mu_0 \left(\frac{T_a}{T_0}\right)^{1.5} \frac{T_0 + 198.72}{T_a + 198.72} \quad \textbf{(41)}$$

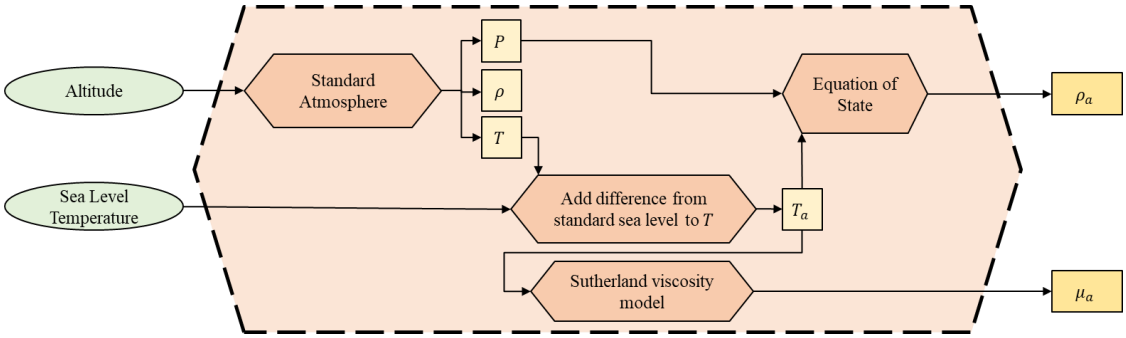

Figure 47: Input-output diagram for the environment model

The environment model assumes a constant value of gravitational acceleration g equal to 32.17 $\frac{ft}{s^2}$. Such an assumption is reasonable for the altitudes at which aircraft operate.

9.3.3 Kinematic Model

Aircraft kinematics were modeled with only two degrees of freedom, considering motion vertically and horizontally. This simplification was deemed acceptable, as preliminary performance estimation typically assumes that the aircraft motion is constrained within a plane. More degrees of freedom could be added in future. A graphical illustration of the coordinate system used for performance estimation and mission modeling is presented in Figure 48. Eqn. (42) and Eqn. (43) relate the vertical and horizontal airspeed components to the airspeed v_∞ through the climb angle γ. A constant horizontally directed wind speed was included within the performance model, leading to the coexistence of two inertial reference frames, one with respect to the ground and the other with respect to the wind. The relative motion of these two frames is illustrated in Figure 48. Eqn. (44) shows how ground speed may be computed from the horizontal airspeed component $v_{h,air}$ and the wind speed v_{wind}. Positive values of v_{wind} denote headwind. In future, the module could be improved by modeling cross winds.

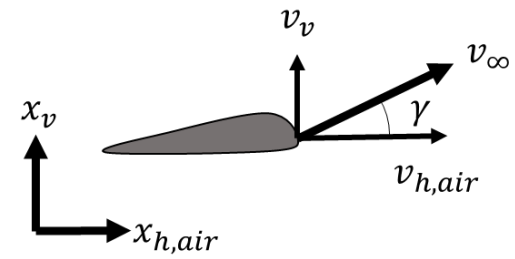

Figure 48: Aircraft model and kinematics in a wind frame

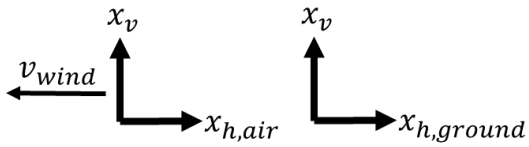

Figure 49: Relative motion of wind frame with respect to ground frame

$$v_v = v_\infty \sin(\gamma) \tag{42}$$

$$v_{h,air} = v_\infty \cos(\gamma) \tag{43}$$

$$v_{h_{ground}} = v_{h,air} - v_{wind} \tag{44}$$

Some additional modeling parameters included within all performance computations were a C_{D0} addition to model unclean configurations, the initial position, time, and battery charge, sea level temperature, and an auxiliary current draw from the main battery for non-propulsive subsystems.

9.3.4 Time Integration Approach

To predict the performance of an aircraft over the course of a mission segment, it was necessary to integrate performance equations with respect to time. To reduce computational effort, the Euler method was used for numerical integration within the performance module methods, despite its crude level of accuracy. Quantities that required integration include battery charge and vertical and horizontal speeds and distances. Horizontal speeds and distances were computed with respect to both the air frame and the ground frame. The integration equations used within the performance methods are depicted

in Eqn. (45) through Eqn. (51). a_v is the vertical acceleration with respect to either inertial reference frame; a_h correspondingly is the horizontal acceleration. The integration interval dt is a user-definable input to each performance computation method. The subscript i implies an initial state. q is the battery cell charge (in Ah), and I_c is the current draw from the cell.

$$v_v = a_v dt + v_{v_i} \tag{45}$$

$$v_{h,air} = a_h dt + v_{h,air_i} \tag{46}$$

$$x_v = v_v dt + \frac{1}{2} a_v dt^2 + x_{v_i} \tag{47}$$

$$x_{h,air} = v_{h,air} dt + \frac{1}{2} a_h dt^2 + x_{h,air_i} \tag{48}$$

$$x_{h,ground} = v_{h,ground} dt + \frac{1}{2} a_h dt^2 + x_{h,ground_i} \tag{49}$$

$$q = I_c \, dt + q_i \tag{50}$$

$$t = dt + t_i \tag{51}$$

Performance analysis results for time-integrated analyses are output as data tables, with rows of values at each integration step. The data table includes the propulsion system state and aerodynamic forces and coefficients. If the aerodynamic analysis was conducted in AVL, the table reports stability derivatives, modes, and neutral point. This table output format makes it especially easy to visualize the vehicle's performance over the course of time. Furthermore, it can be compared against flight test data if such data is gathered later in the design process. As an illustration, Figure 50 visualizes some select vehicle parameters over the course of a steady climb analysis.

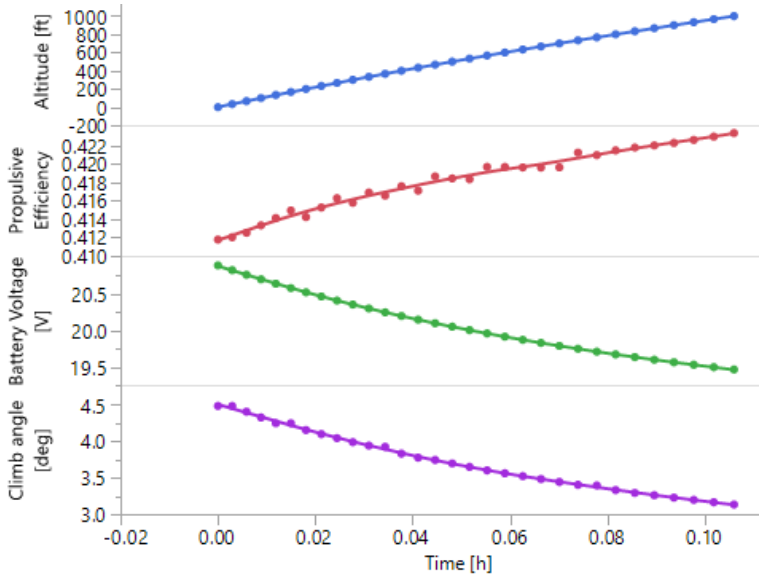

Figure 50: Simulated climb performance predictions vs. time for a generic vehicle model

9.3.5 Modeling Capabilities

The performance module supports the following modeling capabilities:

- Steady Level Flight
- Steady Banked Flight
- Accelerated Level Flight
- Steady Climb
- Steady Glide
- Maximum Speed
- Maximum Rate of Climb
- Takeoff
- Landing Ground Roll
- Catapult or Hand-launch

The assumptions and force balance equations used in these analysis functions reflect those presented by Anderson and Raymer [7] [9]. They are documented in section A.6 of the appendix.

9.4 Performance Module Summary

In short, a performance module was included as part of the unmanned electric flying wing MDA framework to facilitate performance estimation. The module offered functions to model the vehicle's aerodynamic and propulsion parameters over the duration of typical

mission segments. An essential feature within these time-spanning analyses was the numerical integration of battery current to help capture battery discharge behavior.

Future improvement efforts could implement more accurate numerical integration methods and introduce more degrees of freedom into the modeling approach. Additional mission segment models could be developed, or existing models could be generalized to include more situations. Nevertheless, the module as implemented provides a core functionality that is more than sufficient and accurate for preliminary performance evaluation.

CHAPTER 10. FRAMEWORK INTEGRATION

This chapter presents the synthesis and culmination of all previously described work. The automated disciplinary analyses are finally integrated to form the multidisciplinary design analysis framework created to validate Hypothesis 1.0. The programmatic implementation of this framework is discussed first, followed by the presentation of a mission modeling framework created to facilitate mission performance simulation. Finally, verification and validation studies are documented to provide evidence of the framework's utility in addressing the challenges of unmanned electric flying wing design.

10.1 MDA Framework Implementation – the Vehicle Class

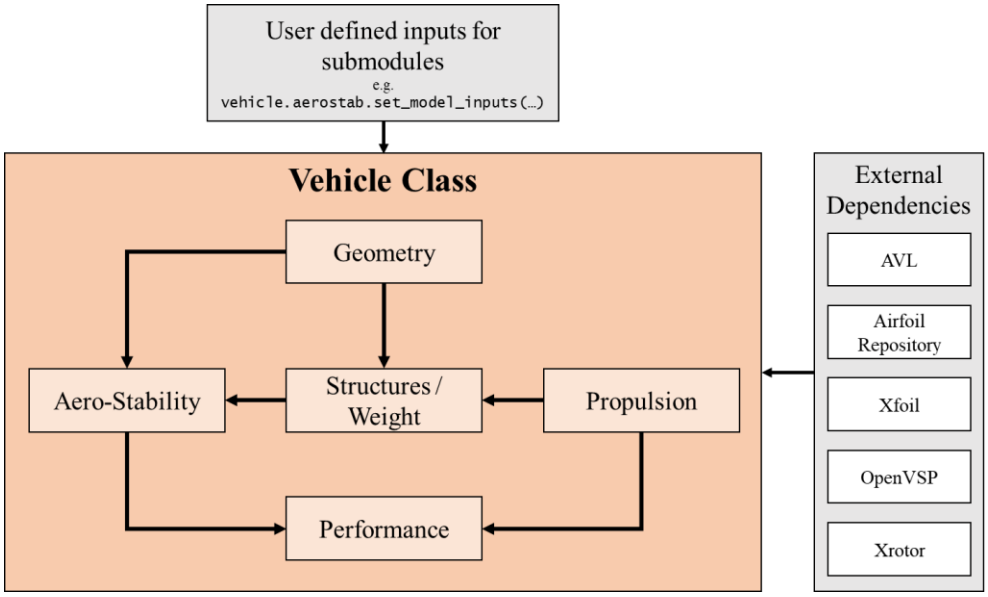

Figure 51: Simplified diagram of vehicle class showing module interdependencies

The integrated MDA framework is instantiated as a programming object known as the vehicle class. Figure 51 shows that the five disciplinary modules (geometry, aero-stability, propulsion, structures, and performance) are subclasses of this object. As a result, disciplinary analyses are separated into compartments for improved organization.

After instantiation of the vehicle class, the user must assign inputs to the five submodules. These input access points are summarized in Figure 52.

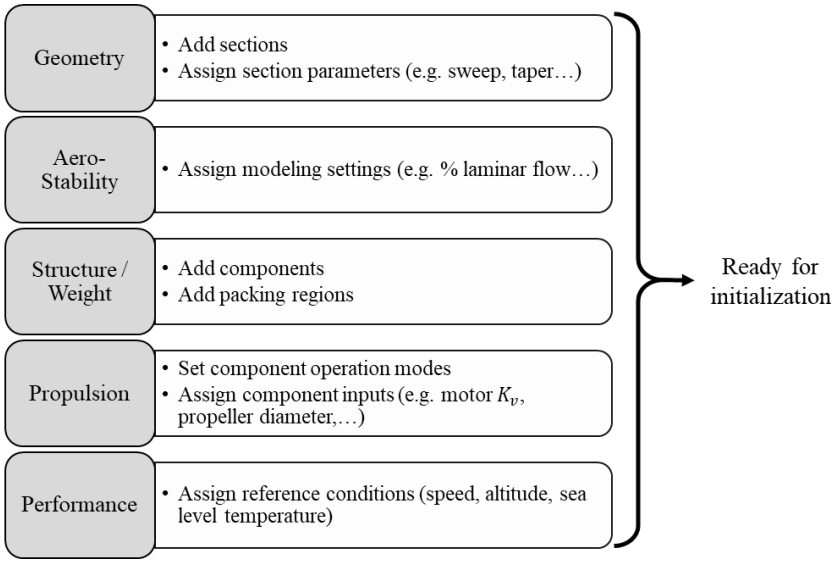

Figure 52: User interactions with the vehicle class prior to initialization

Finally, the modules themselves need to be interconnected as Figure 51 illustrates. These interlinkages between modules are automatically set up by calling an initialization function. It follows the steps shown in Figure 53: First, the geometry class is provided as an input to the structures and aero-stability modules. Then cuboids are packed in the defined packing regions and the propulsion system component mass properties are added to the weight distribution table. In this way, the weight properties are brought up to date. These can then be passed to the aero-stability module, which is itself passed to the performance module along with the propulsion class.

At the conclusion of this initialization procedure, the disciplinary analyses have been connected in a web of input-output relations. What results is a virtual vehicle model with which a user can execute point performance or disciplinary analyses to evaluate the design. Most importantly, the user can define a mission profile using a supporting mission

modeling framework and simulate the vehicle's performance over the course of the mission.

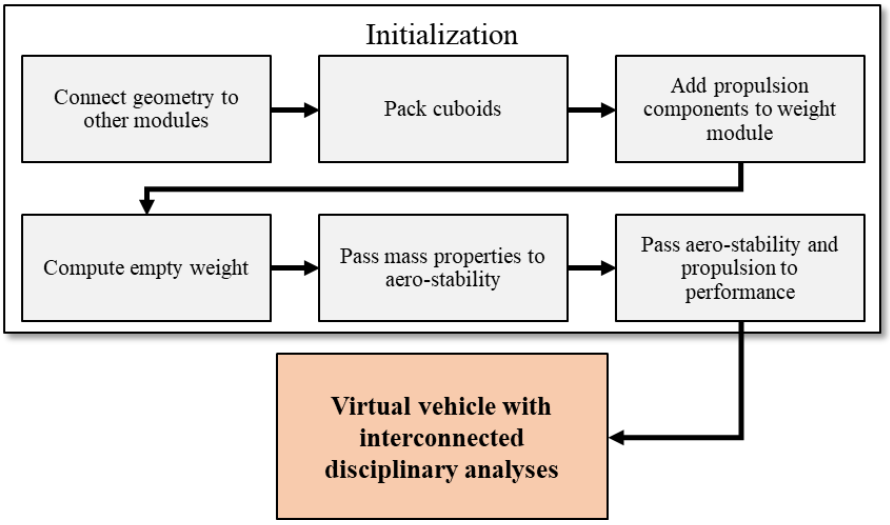

Figure 53: Steps in vehicle class initialization

One of the main disadvantages of the framework in its current form is the required use of the initialization function to reconnect the disciplines if an input on one of them is altered. Forcibly reconnecting all the modules after small changes is inefficient. A future improvement to this approach could employ Python's pass-by-reference architecture to passively update interconnections.

10.2 Mission Modeling Framework

Aircraft are generally designed to perform a specific mission. Mission modeling predicts the performance of the vehicle over the course of a mission profile, i.e. a sequence of mission segments. To that end, a mission modeling framework, separate yet conceptually linked to the MDA framework, was developed to facilitate the assembly of a mission profile and the execution of a mission simulation.

10.2.1 Mission Segments

The basic block of the mission modeling framework is a mission segment class. Distinct mission segments, such as takeoff, climb, loiter, and landing were defined using this class as a template. Each segment has unique inputs, which the user must specify. The object's primary function executes a simulation by passing the mission segment inputs to the appropriate performance module method of a vehicle class. The vehicle class is passed as an added input to the function. This programmatic implementation is depicted graphically in Figure 54.

If desired, the speed input for the mission simulation can automatically change depending on the input vehicle, such that it is set to the speed for minimum power or minimum thrust. In determining this speed, the simulation method relies on the reference drag polar in the vehicle performance module. Moreover, mass components with specific names can be removed from a vehicle class's weight properties module to model a payload drop.

In all, ten separate mission segments were implemented:

- Takeoff
- Steady Climb
- Steady Level Turn
- Cruise
- Steady Glide
- Launch
- Accelerated Level Flight
- Loiter
- Loiter until discharged
- Landing Ground Roll

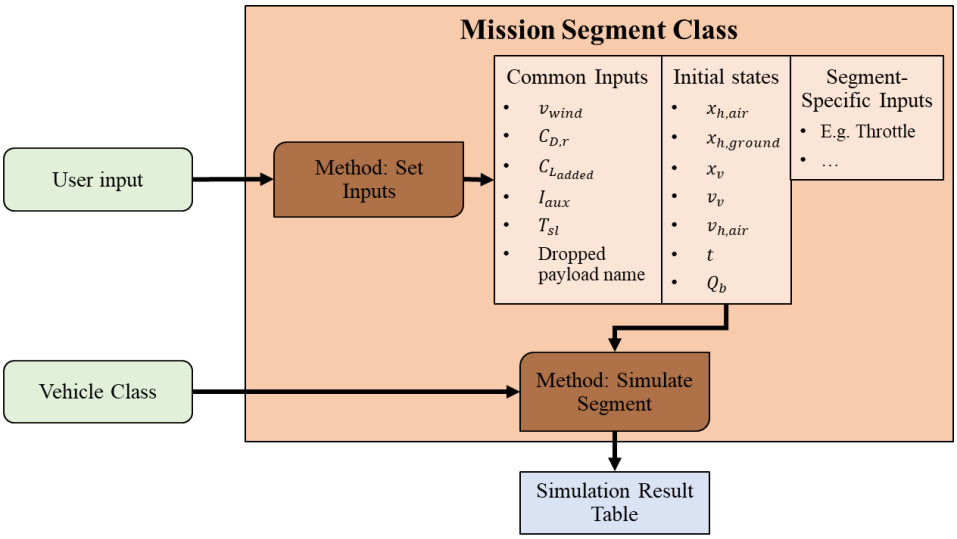

Figure 54: Main methods and attributes of the mission segment class

10.2.2 Mission Profile

The second class supporting the mission modeling framework is a mission profile class. This object is used to assemble a sequence of mission segments in a desired order and execute a simulation over all of them in turn. As shown in Figure 55, the user may add, edit, or remove specific mission segment classes. After setting initial conditions and global inputs, the user executes the primary class method, which simulates the mission profile given a vehicle class as an input. This method simply steps through the simulation methods of each segment in order, feeding the final state of one segment to the initial state of the next. The output is a data table documenting the vehicle's aerodynamic and propulsion system states at each time step in the numerical integration. In this manner, the user's interaction with the framework can be drastically reduced, and it is possible to simulate multi-segment missions with ease.

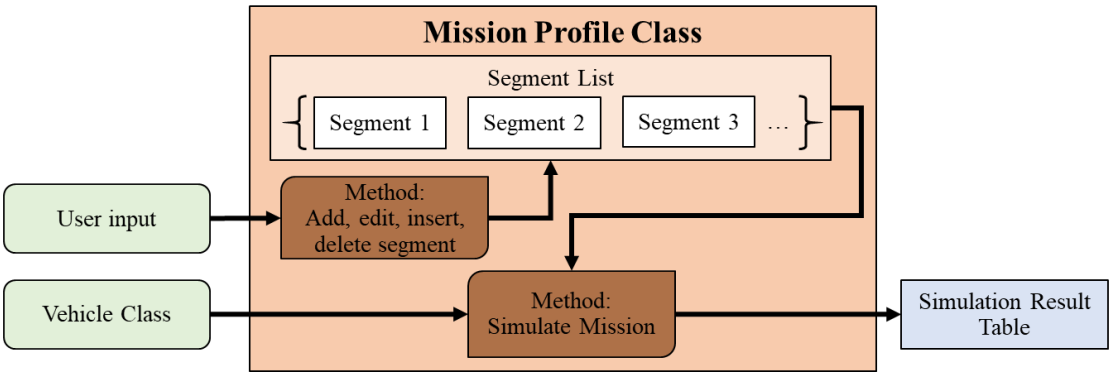

Figure 55: Main methods and attributes of the mission profile class

The mission profile requires altitude continuity between segments. Graphical visualizations of the mission's altitude profile can be generated automatically for verification, as exemplified in Figure 56. Speed continuity, by contrast, is not enforced. A potential area for future work would be to develop a method that automatically inserts acceleration or climb segments to ensure altitude and speed continuity within the profile.

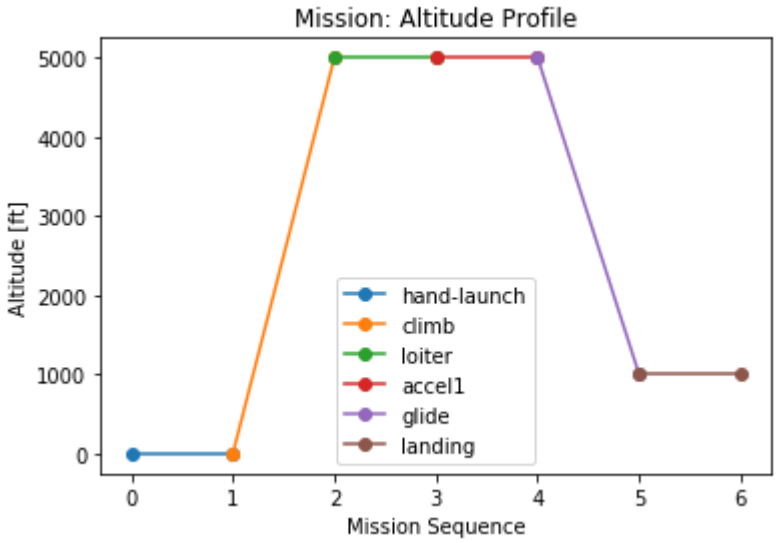

Figure 56: Example altiude profile autogenerated by the mission profile class

In brief, the mission profile class and supporting mission segment object constitute a separate code framework focusing on organizing mission input parameters and streamlining mission modeling efforts. The mission framework takes vehicles as inputs and outputs mission performance for further study. As such, it constitutes an essential

companion piece to the unmanned electric flying wing MDA framework by providing a convenient interface for the performance module.

The major bottlenecks in mission modeling are the throttle-prescribed propulsion system analysis employing Xrotor and the estimation of stall speed and $C_{L_{max}}$ through AVL and Xfoil. Unfortunately, apart from replacing these disciplinary analyses with more rapid surrogates, little could help remedy these bottlenecks, since the external programs already perform at an efficient level.

10.3 Experiment: Framework Verification

After integrating the modules within the framework, it was essential to verify the multidisciplinary analysis workflow. To that end, an experiment was conducted in which predicted performance metrics of a vehicle modelled within the framework were compared against the recorded performance of the actual vehicle. If the predicted metrics matched the actual values to within an acceptable error, it would indicate that the analysis workflow provided realistic performance estimates.

The vehicle selected for the verification test was a flying wing design documented by Karakas et al [66]. The authors reported the design and flight testing of a hand-launched flying wing for surveillance missions. The flying wing configuration had been selected due to its potential reduced drag and weight, which would lead to increased time-on-station. Endurance was targeted at 3 hours. Stall speed needed to be under 40 ft/s for feasible hand-launch.

10.3.1 Procedure

Design parameters for the tailless vehicle were extracted from the article to develop a geometric model of the wing, as depicted in Figure 57. The motor, propeller, and battery

were all modeled using parameters documented within the article and repeated in Table 5. The propeller model relied on the parametric Xrotor analysis, while the battery discharge behavior was based on generic parameters for lithium ion batteries.

Some model alterations were necessary. First, the actual vehicle was not a pure all-wing aircraft, as it had a fuselage pod. The effect of this pod was approximated by adding a constant value to the drag coefficient. Nor could the actual Fauvel 14% airfoil be modeled because the corresponding coordinate file led to several failed Xfoil analyses. Instead, the geometrically similar MH78 airfoil was substituted.

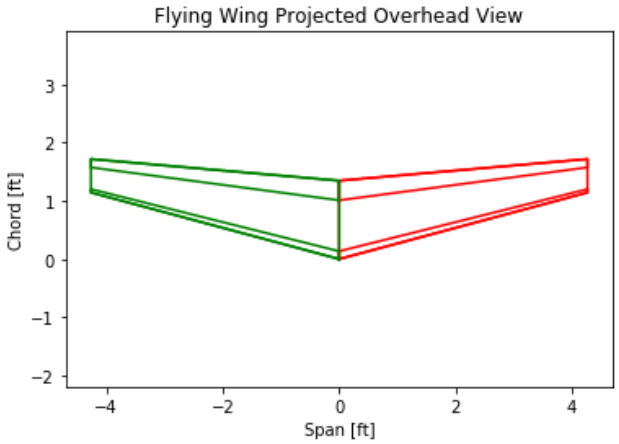

Figure 57: Geometry of the ITU tailless aircraft

Table 5: Design parameters of the ITU tailless aircraft

Parameter	Value
Planform Area	8.186 ft^2
Aspect Ratio	8.89
Sweep	15°
Taper Ratio	0.4265
Airfoil	Fauvel 14% (MH78 in model)
Propeller	12”x8”
Battery	8Ah 16C Li-Ion cells, 5 series, 4 parallel
Motor	465 K_v, 52A max

Once the model was developed and initialized within the MDA framework, performance metrics could be evaluated. Two metrics were considered: stall speed and endurance. Stall speed was estimated at standard sea level conditions using the aero-stability module’s analysis capabilities. Endurance at a 1000 ft altitude at a constant 65 ft/s speed was evaluated with the performance module by integrating current over time until the battery cell voltage reached 1.05 times the cutoff voltage.

10.3.2 Observations

The endurance and stall speed estimates obtained with the framework are reported in Table 6, alongside the actual quoted values. The stall speed estimate differed from the recorded value by only 4%. There was a larger discrepancy of 20% in the endurance prediction.

Table 6: Predicted and reported metrics for the ITU tailless aircraft

Parameter	Documented	Predicted	% Difference
Stall Speed [ft/s]	40	38.3	-4.25
Endurance [h]	3.0	3.59	19.4

10.3.3 Conclusions

Due to the several modeling inaccuracies introduced through either lack of data or conscious alteration, it was not expected for the predicted performance to accurately reflect the recorded values. Nevertheless, a relatively close agreement would be taken as an indication that the modeling tools were functioning properly and outputting results on the correct order of magnitude.

The stall speed prediction matched closely, suggesting the aero-stability module was accurately predicting the maximum lift coefficient. By contrast, the framework overestimated time-on-station by 20%. Such an overestimation is not wholly nonsensical, as the end condition used in actual flight tests was not documented in the article. Moreover, the exact characteristics of the battery could not be modeled accurately. In that light, the 20% discrepancy was sufficiently low to suggest the underlying code was still correctly modeling the physics.

Unfortunately, there were no additional performance metrics with which other parts of the MDA framework could be verified. Nevertheless, the promising comparison results for stall speed and endurance indicated that the framework had been correctly integrated

and that the modules used in the multidisciplinary analysis workflow yielded realistic predictions.

10.4 Experiment: Computational Time Assessment

The overarching hypothesis claimed that using the framework would lead to significant savings in analysis time. In order to test this statement, an experiment was performed to compare the computational efficiency of employing the MDA framework as opposed to a manual analysis process. As a case study, the time required to conduct a simple mission simulation was examined.

10.4.1 Procedure

The experiment built off the work accomplished in the previously described verification experiment. The script used to develop a model of the ITU tailless aircraft was extended to include a mission simulation. A mission profile consisting of hand-launch, climb, and loiter until discharged was set up using the mission modeling framework (see Figure 58). This mission reflected the actual mission for which the ITU tailless aircraft was designed. Descent and landing were not included, since the vehicle fell to the ground with a parachute. The performance of the ITU vehicle model over the course of this mission was then evaluated using the mission profile's simulation method.

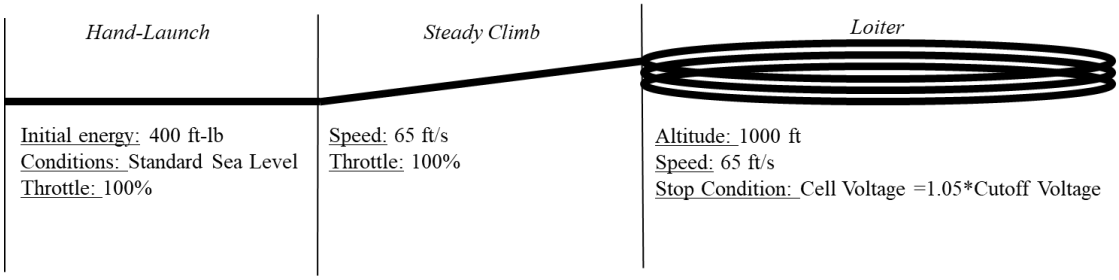

Figure 58: Mission profile used in design space exploration

As part of the experiment, the total time required to complete the mission simulation was measured. Moreover, the total number and average time of external program

executions was quantified. External program executions included automated analyses in Xrotor, Xfoil, AVL, and OpenVSP. The total number of times these programs were called depended largely on the number of integration time steps. With these measurements, it was possible to decompose the total execution time and determine the equivalent number of manual analyses that would have been required to reproduce the result.

Afterwards, the duration of a manual analysis in each external program was estimated by recording the time it would require a human analyst to perform the keystrokes and clicks to obtain a result. This time estimate was ultimately compared against the recorded value for an automated analysis.

10.4.2 Observations

Table 7 documents the execution time measurements for the mission simulation. The MDA delivered a result in 47 seconds. Repeated executions of the analysis script indicated that there was little variation in the total analysis time from run to run.

As expected, the calls to external programs constituted the bulk of the analysis time. Only 3 seconds were required for assigning inputs, passing data from module to module, and performing other more basic computations.

The total estimated time for a comparable manual analysis summed up to 107 minutes. This estimate was obtained by multiplying the total analysis executions used within the mission simulation by the estimated time to perform an individual analysis. In practice, the first manual analysis typically requires more time than any additional analyses, since certain inputs would not need to be re-entered for multiple executions. These time savings were not reflected in the estimate. However, neither was the time required to parse output results and pass them to future analyses, which would likely require more time overall than

the actual analysis itself. Therefore, the time estimates for manual analysis should be viewed as significant underestimates of the actual time required.

Table 7: Analysis execution time recordings

	Number of Executions	Time Per Analysis [sec]		Total Time [sec]	
		Automated	Manual	Automated	Manual
OpenVSP Parasite Drag	6	1.02	10	6.13	60
AVL Trim Solution	52	0.20	60	10.54	3120
Xfoil Drag Polar	32	0.39	60	12.50	1920
Xrotor Analysis	33	0.41	40	13.45	1320
Cuboid Packing	1	1.39	-	1.39	-
Additional tasks	1	3.21	-	3.21	-
			Sum:	**47.22**	**6420**

10.4.3 Conclusions

The time measurements indicated that for a relatively simple mission profile consisting of hand-launch, climb, and loiter, a manual analysis process would take at least 130 times longer than the automated process offered by the framework. While the framework provided a solution in under a minute, the manual process would take over an hour and a half at the very least.

In practice, no human analyst would ever execute the external analysis programs as often as the framework can. Rather, the analyst might execute it a few times to obtain some datapoints to develop a regression, such as a quadratic drag polar model. But even if such regressions were used in place of the actual analysis program, developing them would still require a significant amount of time. Moreover, if any of the design variables were to change, the regressions would need to be developed anew. Therefore, the comparison this experiment provides still illustrates the time differences between an automated and manual analysis approach

In brief, the results of the experiment proved decisively that the MDA framework offered immense analysis time savings as compared to the alternative of a manual

analysis procedure. It did so with enhanced, rather than diminished, analysis fidelity, since the framework directly calls external analysis software rather than employing regressions. Therefore, the hypothesis that employing an MDA framework would allow designers to uncover feasible designs more quickly with more fidelity is validated.

10.5 Demonstration: Design Space Exploration

After proving the framework's efficacy in comparison to a manual analysis approach, a follow-on study was executed to demonstrate the framework's effectiveness in design space exploration. This study further highlighted the MDA framework's automated workflow that allows a user to almost effortlessly investigate and compare vast numbers of different designs.

10.5.1 Procedure

The design space exploration was conducted by sequentially modeling the mission performance of vehicles with different design variable inputs. The baseline vehicle and mission were the same as used within the computational time assessment experiment. Seven geometric and propulsion system design variables were selected to perturb within the exploration. Several thousand input combinations were defined using a design of experiments (aka DoE). The parameter ranges within the DoE are documented in Table 8. Parameters not altered in the DoE were left at their estimated value for the baseline vehicle. Additional assumptions and procedures included:

- The battery packing code placed 20 batteries as far forward in the wing as possible.
- Empty weight fraction scaled linearly with reference area.

- The masses of ESC, motor, propeller, and wire were estimated using the propulsion module's built-in component mass regressions. These regressions extrapolated in some cases.
- The motor and propeller were positioned at the rear of the root chord, while the ESC was placed at the center of the root chord.
- The maximum rated current for motor, ESC, and wiring was initially set to 52A. During framework initialization, the maximum current flow was computed for a 100% throttle static test. If this predicted maximum current exceeded the default value, the rating was adjusted to equal 1.5 times the predicted maximum. This adjustment would change the mass of the propulsion system.

Table 8: Parameter ranges for exploration study

Parameter	Min	Max
Pitch-to-Diameter Ratio	0.50	0.80
Propeller Diameter [in]	8.0	14.0
Motor K_v [RPM/V]	300	1200
Aspect Ratio	6.0	12.0
Planform Area [ft^2]	6.0	10.0
Sweep [^{o}]	10.0	25.0
Taper Ratio	0.30	0.80

As in the previous experiment, the MDA framework predicted the performance of each vehicle over the course of the baseline mission shown in Figure 58. Performance metrics were saved for each case, along with mass properties, the aircraft's aero-stability characteristics at trimmed loiter, and the propulsion system state at the beginning of loiter.

In all, about 5900 cases in the DoE were run. The predicted metrics were saved as a table for post-processing in the statistical software JMP. Of the several hundred recorded responses, five were selected as criteria by which vehicles would be compared: static margin, stall speed, time-on-station, time-to-climb, and mass.

10.5.2 Results Filtering

The primary exploration exercise consisted of filtering out unsatisfactory designs. Vehicles exhibiting any of the following features were eliminated from the design space:

- Modes with positive real part
- Static margin outside the range of 6-15%
- Stall speed greater than 40 ft/s
- Time-to-climb greater than 3 minutes
- Mass greater than 9 kg
- Wingspan greater than 9.84 ft (3 m)

Figure 59 shows a constellation plot of the modeled vehicles before filtering, while Figure 60 shows it after filtering. The constellation depicts the performance metrics plotted as functions of each design variable. A clear reduction in feasible designs due to filtering was apparent when comparing the constellations before and after filtering. The apparent discrete response of stall speed was the result of using the bisection method with a relatively high termination tolerance.

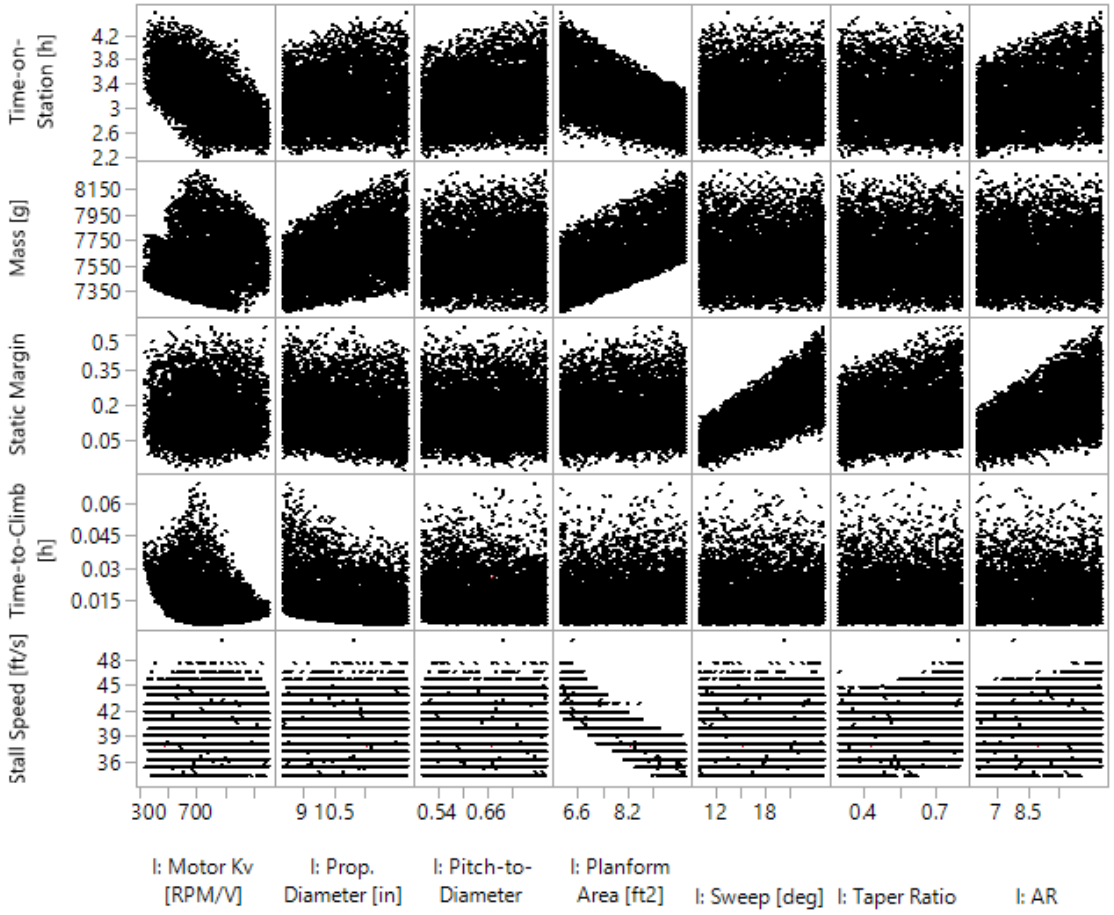

Figure 59: Performance parameters vs. design variables prior to filtering

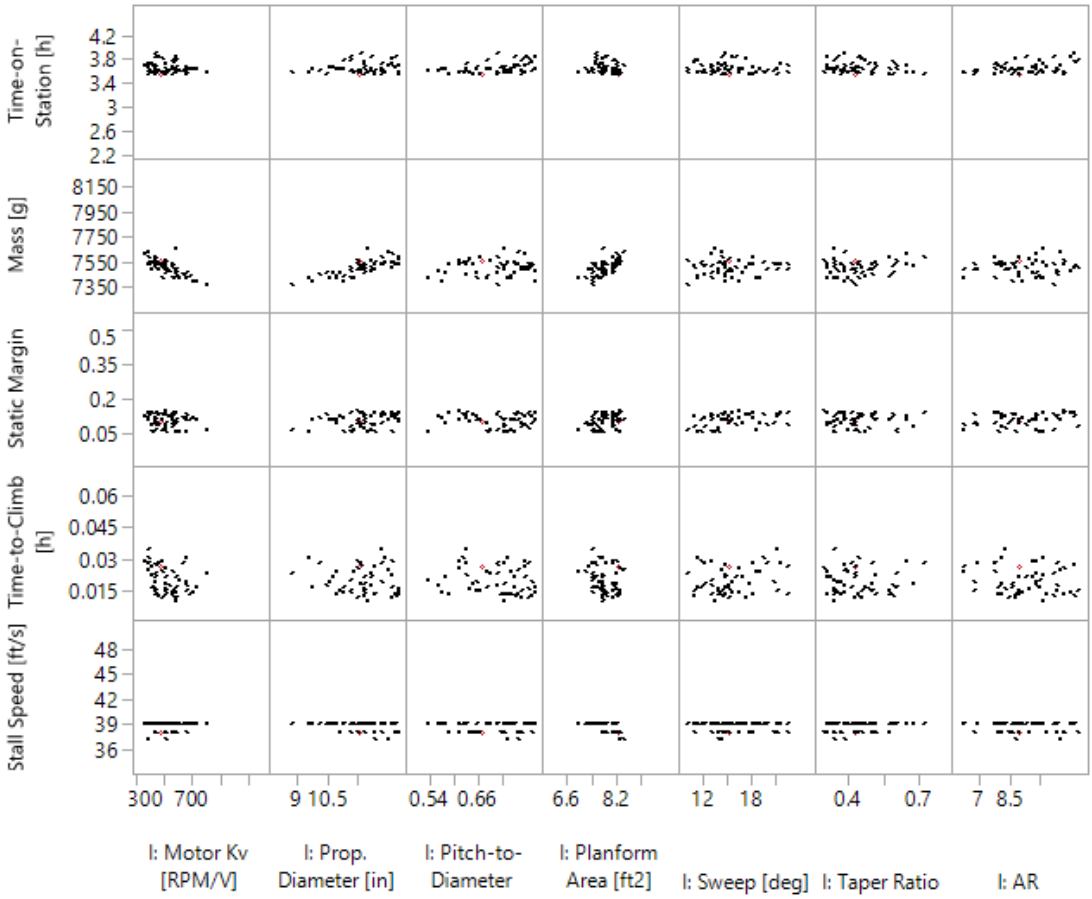

Figure 60: Performance parameters vs. design variables after filtering

After filtering, time-on-station was chosen as the most important metric, and the vehicle with the most endurance that satisfied all the previous constraints was identified from the remaining design set. This design, which will be called the "filtered optimum", differed little from the baseline in terms of propulsion system parameters, as can be seen in Table 9. The major change was the aspect ratio, which was 30% larger than the baseline. The difference in planform shapes is illustrated in Figure 61. Table 10 shows that the geometry tweaks led to a 10% predicted improvement in time-on-station, and a 50% increase in static margin. Stall speed, time-to-climb, and mass were mostly unchanged from the baseline.

This result illustrated the framework's utility in identifying attractive vehicles by quickly evaluating a large set of designs. A time-intensive manual analysis approach would not have been capable of finding such an improved design in any acceptable amount of time.

Table 9: Input parameters of baseline and filtered optimum vehicles

Parameter	Baseline	Filtered Optimum	% Difference
Pitch-to-Diameter Ratio	.667	0.7086	6.236882
Propeller Diameter [in]	12.0	12.14	1.166667
Motor K_v [RPM/V]	465	437	-6.02151
Aspect Ratio	8.89	11.61	30.59618
Planform Area [ft^2]	8.186	7.545	-7.83044
Sweep [^{o}]	15.0	14.64	-2.4
Taper Ratio	0.4265	.3778	-11.4185

Figure 61: Geometry comparison: left is baseline; right is filtered optimum

Table 10: Performance metrics for baseline and filtered optimum vehicles

Parameter	Baseline	Filtered Optimum	% Difference
Static Margin	0.10	0.147	47
Stall Speed [ft/s]	38	39.6	4.210526
Time-to-Climb [h]	0.02738	0.02780	1.533966
Mass [g]	7573	7520	-0.69985
Time-on-Station [h]	3.588	3.957	10.28428

10.5.3 Sensitivity Analysis

As a final part of the demonstration exercise, surrogate models of the five performance metrics were generated for sensitivity analysis. A surrogate model is a computationally cheap function that computes the value of a response variable, e.g. time-

on-station, based on design variable inputs. In doing so, the surrogate replaces the MDA framework's more computationally expensive routine.

Surrogates were developed by "training" a fit model to predict the responses (i.e. performance metric values) for 75% of the successful cases. The trained model was further evaluated by examining its predictions for the remaining 25% of the data points. Artificial neural networks were employed as the fit's functional form because they could better model the nonlinear response of mass due to maximum rated current flow. Ultimately, very strong fits were obtained for all five performance metrics using these regression analysis techniques.

For further insight into trends, the partial derivatives of these surrogates were plotted to reveal sensitivities of the performance metrics with respect to the design variables, as seen in Figure 62. These sensitivities not only offered a way of verifying results by checking trends against intuitive expectation, but also indicated which parameters impacted overall performance the most. For instance, taper ratio seemed to only affect static margin and stall speed. Propulsion parameters, by contrast, affected almost all performance parameters. These sensitivities demonstrate that the framework is indeed multidisciplinary, since the parameters of one discipline visibly impacted metrics from other disciplines.

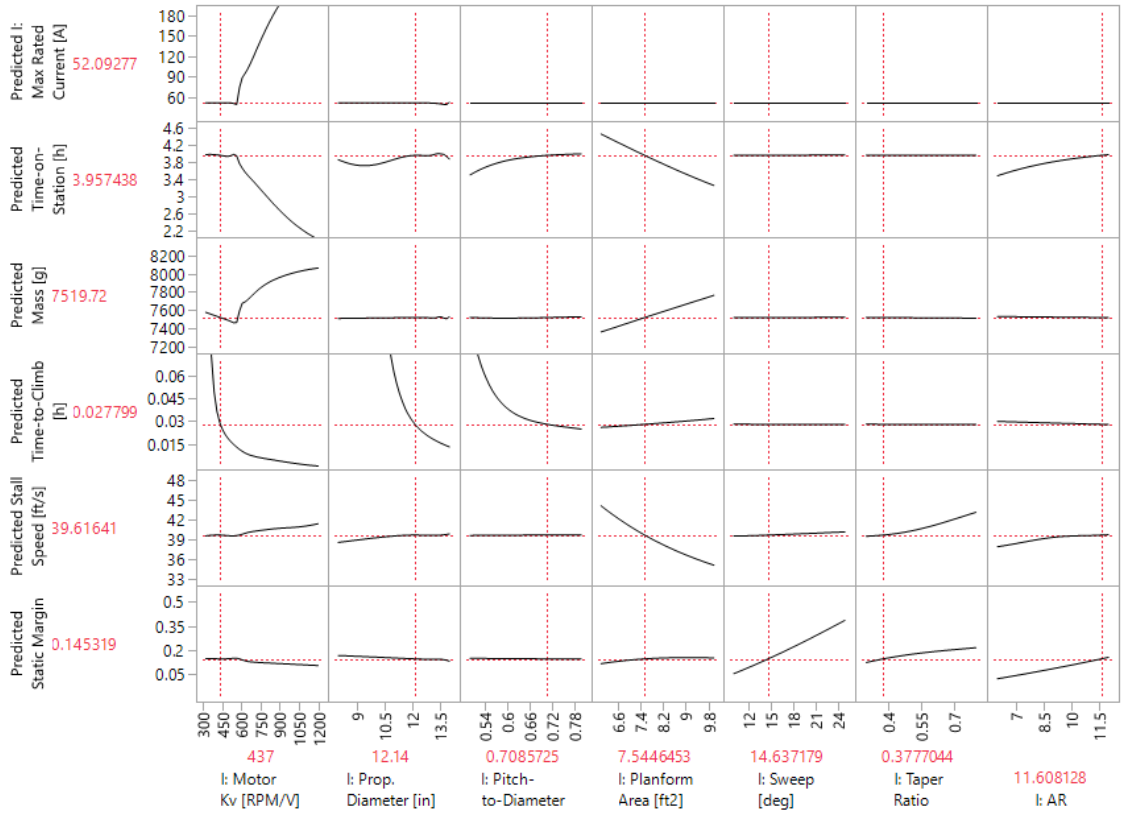

Figure 62: Design sensitivity plots at the filtered maximum endurance vehicle

In brief, the demonstration study proved that the framework facilitates design space exploration and sensitivity analysis. It further illustrated how these capabilities could be used to identify improved designs by exploring the space around a baseline design. Traditional manual disciplinary analyses would have been incapable of achieving this result due to the immense time and effort they would have necessitated.

10.6 Demonstration: Robustness Assessment

A final study was conducted to demonstrate the framework's utility in assessing a design's robustness to uncertainties and changes in requirements. The intent was to examine how a fixed design's performance would be affected by perturbing mission-related parameters, such as loiter altitude, sea level temperature, and payload current draw.

10.6.1 Procedure

As in the previous design space exploration study, the robustness was assessed by evaluating the aircraft mission performance for several different input parameter combinations. These combinations were defined in a design of experiments.

In an effort to consider more than a single baseline vehicle, the ITU tailless aircraft was not employed in this study. The vehicle used here did not represent any existing UAV platform. Its size and shape were determined mostly arbitrarily, rather than through a rigorous sizing process. As such, the results of this study should be regarded as purely illustrative rather than practical. The geometry selected for examination is depicted in Figure 63. The design employed twin propellers to demonstrate the propulsion module's ability to model multiple parallel drive systems. The design parameters chosen for the vehicle are summarized in Table 11. The vehicle's mission, pictured in Figure 64, was in rough outline the same as for the ITU tailless aircraft. Modeling parameters were varied in a DoE with ranges documented in Table 12. The impact of battery cell age was modeled through a linear reduction in capacity. In all, roughly 1800 cases with unique parameter variations ran successfully.

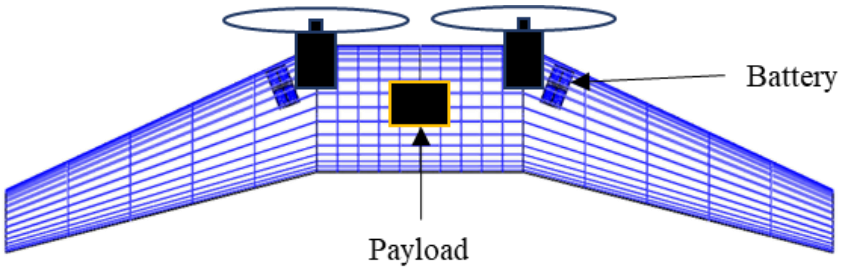

Figure 63: Vehicle configuration used in the robustness assessment

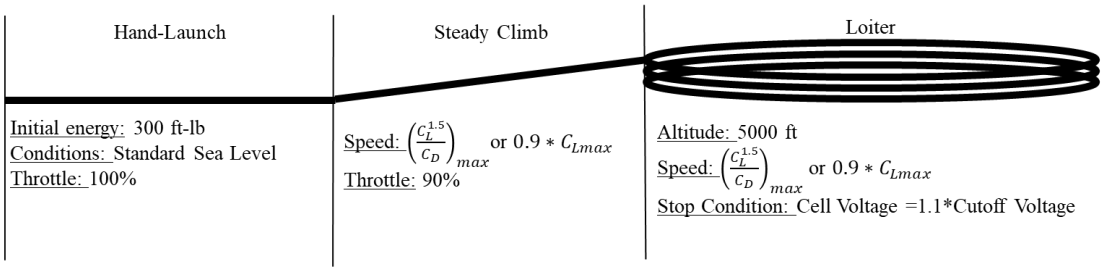

Figure 64: Nominal mission profile used in the robustness assessment

Table 11: Vehicle design parameters used in the robustness assessment

Parameter	Value
Payload Weight	5 lb
Empty Weight Fraction	0.3
Planform Area	8 ft^2
Aspect Ratio	8
Sweep	25°
Twist	-5°
Taper Ratio	0.5
Airfoil	Davis B24
Propeller	9” x 6.3”
Motor	1200 kV, 100 A max current
Battery	3S24P 3700mAh LiIon

Table 12: DoE parameter ranges for robustness assessment study

Parameter	Min	Max
Sea level temperature [°F]	32	100
Empty weight fraction [%]	20	40
Secondary current draw [A]	0.0	7.0
Loiter altitude [ft]	4000	6000
Cell age [Cycles]	0	250

10.6.2 Sensitivity Analysis

The computed responses for the successfully run cases were used to create response surfaces of the four metrics of interest: static margin, time-on-station, time-to-climb, and hand-launch drop height. Strong fits could be obtained using least-squares regressions. As in the previous demonstration study, the fits were developed using 75% of the data for training and 25% for validation.

The sensitivities of performance metrics with respect to the input variables could be conveniently plotted using the partial derivatives of the developed surrogates. An example of such a plot generated in JMP is shown in Figure 65. Several insights might be drawn from these sensitivities. For example, one can see a strong sensitivity of static margin on empty weight. This arose because empty weight was assumed to be centered at the center of area of the flying wing planform, which in this case lay behind the neutral point. An additional constraint on empty weight fraction came from the launch drop distance: Achieving a reasonably small drop distance required an empty weight fraction verging on the lower examined bound of 20%. Some other notable sensitivities arose due to sea level temperature. Colder temperatures reduced the drop distance due to the increased air density (assuming a constant pressure). Similarly, the time-to-climb decreased with temperature. Time-on-station tended to decrease with decreasing temperature due to battery discharge behavior

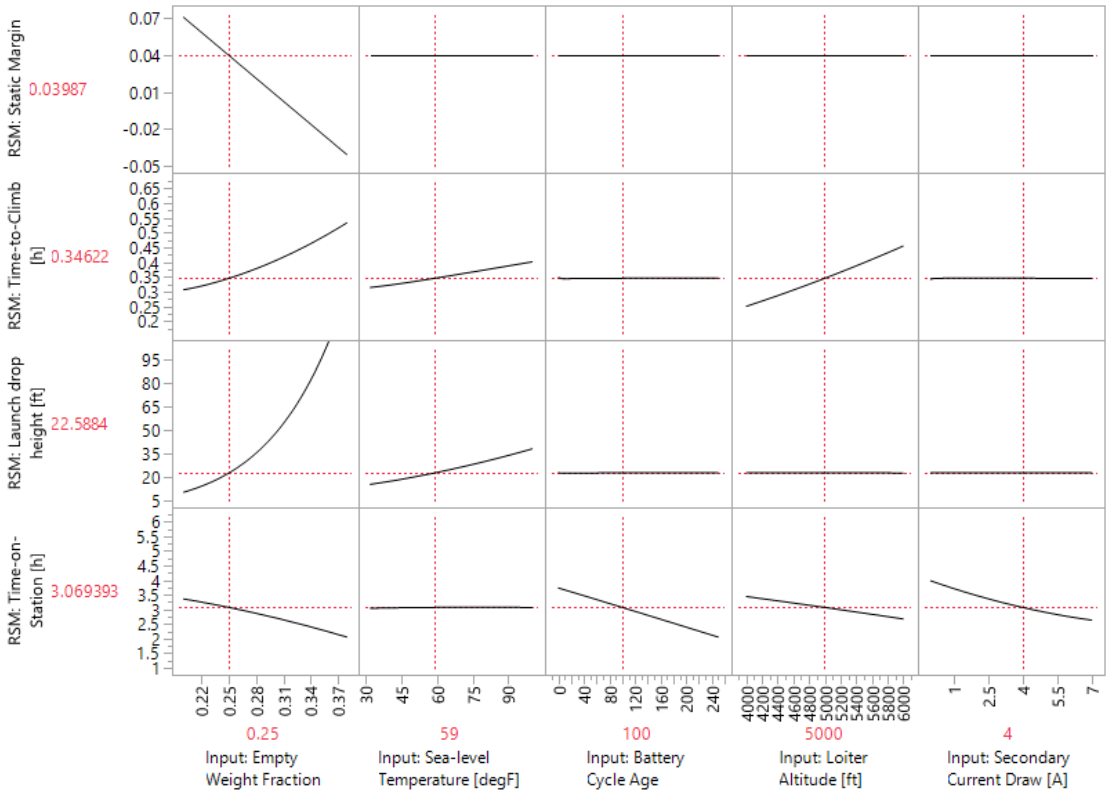

Figure 65: Response sensitivities to robustness parameters

In brief, the MDA framework could readily examine a design's robustness to changes in environmental conditions or requirements. Such a capability is of great value in many scenarios. For instance, it could help evaluate the suitability of an existing UAV platform to a new application. The rapidity with which the MDA framework could provide answers in such design scenarios makes the tool vastly superior to the main alternative, which would be executing a sequence of manual disciplinary analyses.

10.7 Framework Integration Summary

In brief, the multidisciplinary design analysis framework developed within this thesis takes the final form of a Python object containing the previously documented disciplinary modules as subclasses. Interdisciplinary input-output linkages are set up virtually through an initialization function. Mission performance modeling is facilitated through a separate mission modeling framework, with which a user can easily define arbitrary mission

profiles. Finally, the framework's functionality and utility were investigated with two experiments and two demonstration studies. The first experiment verified performance predictions of a flying wing modeled after an existing UAV. The second experiment proved beyond doubt the superiority of the framework over manual analysis approaches with regard to computational time and effort. The first demonstration study displayed the power of the framework's automation capability to explore a large design space and uncover potentially improved designs. The final investigation showed that the framework could assess the robustness of a design to changes in mission requirements or uncertainty in design variables. In all, these studies demonstrated how the MDA framework could capture interdisciplinary couplings and facilitate multidisciplinary trade studies that would be prohibitively time-consuming for a traditional manual design process.

CHAPTER 11. CONCLUSION

11.1 Research Effort Summary

This thesis targeted the broad field of unmanned aerial vehicle design. As a starting point, unmanned aerial vehicles were shown to represent relevant platforms with wide ranges of applications in military and civilian contexts. All-wing aircraft were presented as a notable configuration option, offering benefits in aerodynamics, survivability, and structures. They were also shown to exhibit competing disciplinary interactions that posed challenges in design. Electric power was also discussed as an important contemporary area of research. Consequently, unmanned electric flying wings emerged as a promising UAV configuration with unresolved design challenges. These challenges related primarily to strong interdisciplinary couplings, which a traditional, manual, discipline-separating design process could not readily address.

This thesis's overarching research question asked how these couplings could be addressed within design to arrive at improved vehicle concepts. A potential solution emerged from a review of the increasingly relevant field of multidisciplinary design analysis. It was subsequently hypothesized that by developing an MDA framework for unmanned electric flying wings, it would be possible to capture the interdisciplinary couplings and provide previously nonexistent design capabilities.

In order to validate the hypothesis, an MDA framework was developed using object-oriented programming in Python. It was necessary to develop five interlinking disciplinary modules:

Geometry: The geometry module provided tools for quickly modeling symmetric wings made up of trapezoidal spanwise sections.

Structures and Weight: The weight-distribution module included a component mass property table, from which values of center of mass, mass, and mass moment of inertia were automatically computed. Structural mass was modeled heuristically assuming either weight fractions or a monocoque shell design. The module included the capability to model dense, geometrically feasible packing arrangements of battery cells within the wing shape for improved weight distribution estimates.

Propulsion: An original propulsion system analysis module was developed, containing models of propeller and motor performance, ESC mass and efficiency, wire resistance and mass, and battery discharge. These stand-alone models were integrated within a larger propulsion system framework, which could predict the power losses at thrust or throttle-prescribed conditions. The code compared favorably to commercially available software.

Aero-stability: The aero-stability module estimated vehicle lift, drag, stall speed, stability derivatives, control surface deflections, lift distribution, and dynamic modes. This was accomplished through automation wrappers around AVL, Xfoil, and OpenVSP.

Performance: A performance module codified performance equations. This module relied on the others to predict such things as maximum speed, maximum rate of climb, range, and endurance.

The five modules were combined and interwoven to form the MDA framework. A separate mission definition framework was developed to facilitate arbitrary mission modeling. Finally, the capabilities of the combined MDA and mission framework were documented in three studies.

11.2 Hypothesis Evaluation

The overarching hypothesis of this research effort stated that a multidisciplinary design analysis framework would capture the strong interdisciplinary interactions inherent in flying wings, and in doing so aid their design (see Figure 66). This section summarizes the features by which the MDA demonstrably supports unmanned flying wing design in ways that were previously not possible.

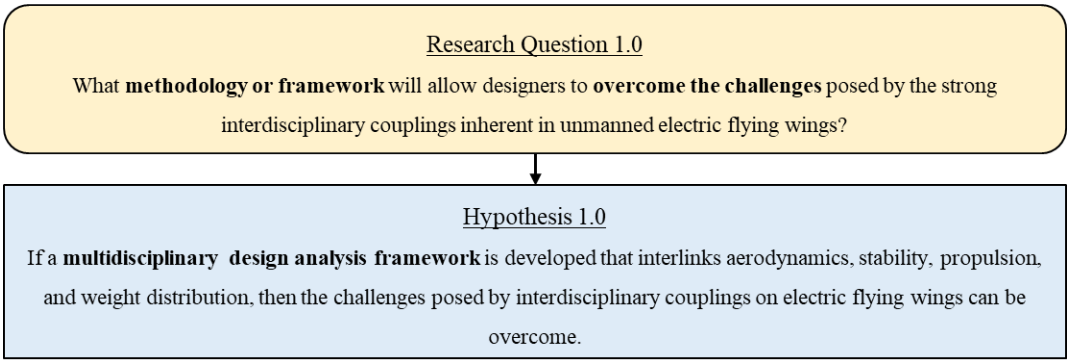

Figure 66: Overarching research question and hypothesis

First, the framework permits automation. Disciplinary and multidisciplinary analyses can be iteratively executed without manual intervention. This capability enables design space exploration and surrogate modeling, which can help identify optimized designs from a baseline in a drastically reduced timeframe. It also facilitates quick trade studies and sensitivity analyses, showing how design variables for one discipline impact the others.

In addition, the framework performs the automated analyses rapidly. Most disciplinary analyses execute in fractions of a second. Some select analyses are more time-intensive, particularly stall speed estimation. A quantitative experiment demonstrated that a mission simulation employing the MDA framework could be executed two orders of magnitude faster than by using a manual approach. This automation capability enables analysts to explore large design spaces in reasonable amounts of time.

Furthermore, the design environment drastically reduces workload and development time by automatically setting up connections between disciplines. It is no longer necessary for one subteam to wait for the results of a manual analysis conducted by a different subteam. A single person becomes able to analyze vehicle performance for arbitrary inputs, thereby reducing the time and resources required for design studies.

Moreover, the framework offers various ways of facilitating data visualization. Data can be output in an ordered table format that is readily opened and visualized in analysis software such as Excel, JMP, Matlab, or Tableau. The disciplinary modules also offer simple sketching capabilities for verification of specific parameters. One important feature is the automated 3D modeling capability within the geometry module: with a single command, the user can generate a CAD model that would normally take several minutes to create by hand.

Additionally, the framework enables simultaneous aerodynamic and stability analysis that reflects the significant impact that trim requirements have on aerodynamic characteristics. In considering this essential interaction, the MDA framework can be used to sift out many infeasible designs, as was demonstrated in a validation study.

Similarly, the framework includes improved weight estimation routines. This capability is made possible through the cuboid packing routine, which itself constitutes the most original contribution developed as a part of this research endeavor. This code delivers geometrically feasible packing arrangements of cuboid components within a wing-shaped volume. It was developed to determine realistic battery placements and improve center of gravity estimation. However, the code is not limited to this application, and could be used

in a variety of other contexts where the goal is to densely pack components represented as cuboids within an extruded contour.

Finally, the propulsion system module represents a valuable contribution to electric aircraft design. The module performs on par with existing commercially available software in terms of accuracy while offering several key benefits. For instance, the models employed for each component can be easily switched out in favor of other analyses or surrogates, making the modeling environment transparent and flexible. The module further enables automation at variable levels of fidelity.

In summary, the MDA framework offers a variety of design aids targeted specifically at unmanned electric flying wings that were previously unattainable in a manual, discipline-separating design process. Briefly enumerated, these contributions are:

1. Fast, automated multidisciplinary analyses
2. Ease of use and limited analysis setup time
3. Automated 3D Modeling
4. Data visualization support
5. Cuboid packing and improved mass property estimation
6. Variable fidelity propulsion system analysis
7. Simultaneous stability and aerodynamic analyses
8. Mission modeling facilitation

In conclusion, these features and capabilities provide overwhelming evidence that the framework does deliver on its promise of significantly improving the design process for unmanned electric flying wings by considering multidisciplinary interactions. Consequently, Hypothesis 1.0 is ultimately validated. With further evaluation and

application of the framework to real design problems, it is expected that both the framework and its underlying modules will continue to prove their effectiveness as aids in design, not merely for unmanned electric flying wings, but for UAVs in general.

11.3 Future Work Areas

While fully functional and effective, the framework still possesses several deficiencies that additional work could ameliorate. These improvement and expansion opportunities are summarized in Table 13.

Table 13: Future work areas

Geometry
• Enable curved edges • Allow control surfaces to only span part of a section • Introduce more modelling features, such as for ducts, pods, or landing gear
Aero-Stability
• Model more trim situations and incorporate them within performance estimation • Export animations of dynamic behavior • Speed up computations (parasite drag, $C_{L_{max}}$) • Allow control surfaces to balance yawing moment
Structure and Weight
• Develop additional structural weight estimation methods • Include basic structural analysis capabilities • Translate the cuboid packing code to a compiled language for greater speed • Account for twist and taper ratios greater than 1
Propulsion
• Develop faster propeller models or surrogates of Xrotor • Integrate Xfoil analyses within parametric Xrotor propeller model – or add aerodynamic properties for different blade airfoils • Include heating models and better estimates of parameters controlling higher-order effects • Develop a parametric model for ducted fans • Include an automation wrapper for Ducted Fan Design Code • Increase the electronic speed controller modelling fidelity • Develop a more flexible power-modeling framework, able to include multiple separate power systems • Add more internally stored battery model parameters • Add a generic battery age effect model
Performance
• Employ more accurate numerical integration techniques • Add more degrees of freedom to the performance equations • Add more equations, e.g. for pull-up maneuvers • Auto-generate v-n diagrams • Model cross winds
Integrated Framework
• Eliminate or reduce the need for the initialization method • Provide a link between the weight/structures module and aerodynamic load distribution
Mission Modeling Framework
• Add more segments • Increase computational efficiency
General
• Continually perform and document verification and validation studies • Develop an aircraft sizing environment around the MDA framework

APPENDIX

A.1 History of Flying Wings

In the 1930s and early 1940s, the German pilots Walter and Reimar Horten, often referred to collectively as the Horten Brothers, were among the first to design, build, and fly large sweptback flying wings [67]. Their work culminated in the design towards the end of WWII of a jet-powered bomber prototype, the Ho 229 (see Figure 67).

As the Horten Brothers developed their designs in Germany, Jack Northrop pushed the flying wing concept in the Unites States. His corporation's efforts in the 1940s produced the XB-35 all-wing bomber and its jet-powered sibling, the YB-49 [68] (see Figure 68). The unconventional appearance of these flying wings immediately captured public imagination. However, Northrop's all-wing aircraft suffered from stability problems and other technical difficulties and could not compete against the faster and larger B-36. While there was some hope of developing the concept further as a reconnaissance vehicle, the US government ultimately cancelled the flying wing project. At the time, it appeared that the configuration had been permanently shelved.

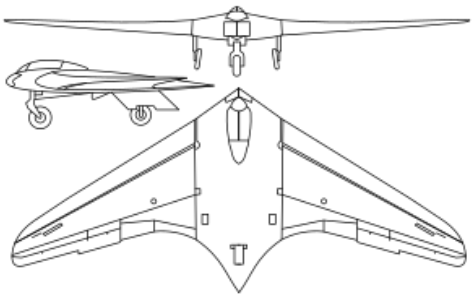

Figure 67: Three-View of Ho-229 [69]

Figure 68: Northrop YB-49 [70]

Not until the advent of fly-by-wire technology and stability augmentation in the 1970s did designers revive the all-wing concept [8]. These enabling technologies made it possible to overcome the stability and handling difficulties of flying wings. The simultaneous concern within military circles that conventional aircraft were becoming too susceptible to advanced air defenses further contributed to a renewed interest in the inherently stealthy all-wing configuration [12]. Thus, the flying wing concept experienced a resurrection, with its new pinnacle visible in the development throughout the 1980s of the iconic Northrop-Grumman B-2 Spirit (shown in Figure 69). More recent unmanned designs continue to demonstrate the configuration's appropriateness for stealth and reconnaissance missions (see Figure 70, Figure 71, and Figure 72)

Figure 69: Northrop Grumman B-2 Spirit [71]

Figure 70: Northrop Grumman X-47B UCAS [72]

Figure 71: Boeing X-45 [73]

Figure 72: BAE Taranis [74]

A.2 Packing Background Research – Cutting Stock Problems

The problem of efficiently packing objects within a bounded space is by no means new. Classic examples include the "knapsack problem" and the "bin packing" problem [75]. When considering only two dimensions, the problem of packing smaller shapes within a larger shape is known as the "cutting stock" problem. This discrete optimization problem arises in many areas, such as cutting items out of textiles, sheet metal, and paper sheets. The problem may be stated as follows, though many variations exist: Given are stock sheets of some shape and a set of smaller shapes. The smaller shapes need to be cut out of the stock sheets. The objective is to cut out all the small shapes using as few stock sheets as possible. The solution will naturally be the case where the small shapes are packed as densely as possible onto the stock sheets without intersecting one another.

Even in its simplest classical form, where the stock sheet is a rectangle and the smaller shapes are rectangles with a common width dimension, the cutting stock problem is difficult to solve due to the potentially immense number of combinations to consider. The computing time required to evaluate all potential solutions even for seemingly small problems can approach astronomic scales [76]. In a seminal paper in 1965, Gilmore and Gomory identified a solution procedure for the classical case by applying linear programming and reducing the problem to a series of knapsack problems [77]. However, this solution does not apply to the more complicated case where the stock sheet is irregular.

Subsequent research into the problem has focused on developing various heuristic packing procedures [76] [78]. Two of note are the so-called bottom-left and bottom-left-fill heuristics. The bottom-left algorithm places shapes by repeatedly sliding them down and then left. Starting from the top right, a shape is slid down until it hits either the

bounding sheet or a previous shape. Then it is slid left until a similar collision occurs. This sequence of sliding moves is repeated until the shape can no longer be moved either down or left without intersecting with another piece. The bottom-left-fill algorithm is an improvement to the bottom-left method, since it fills in voids. Illustrations of these two heuristics are included in Figure 73, which is borrowed from the paper by Hopper and Turton [79].

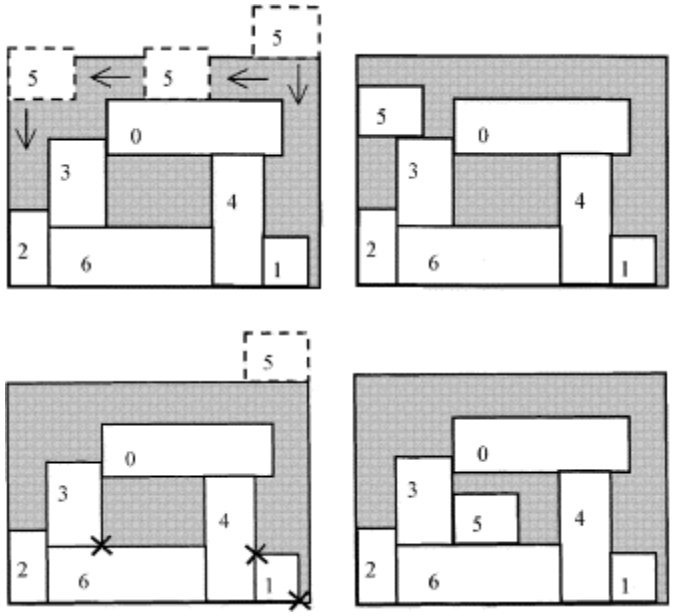

Figure 73: Graphical illustrations of bottom-left (top picture) and bottom-left-fill (bottom picture) heuristics [79]

With the advent of metaheuristic optimization algorithms such as simulated annealing and genetic algorithms, a new solution methodology for cutting stock and bin packing problems emerged. The approach hybridizes a heuristic packing algorithm with a metaheuristic optimization algorithm. The heuristic algorithm generates packing solutions based on some input parameters, like the sequence of items to place or their orientation. Then the metaheuristic optimization algorithm selects, mutates, or otherwise propagates an initial set of input populations to successive generations of increasingly better solutions.

Genetic algorithms are the most popular choice of metaheuristic for this hybridized approach, as seen in [80] [81] [79] [82] [83] [84] [75].

A.3 Parametric Propeller Model Documentation

This section describes the development of the parametric propeller model used in the propulsion module. It discusses how propeller blade geometries for an Xrotor input file are generated automatically from diameter and pitch inputs, and documents how generic blade airfoil aerodynamics were estimated.

A.3.1 Blade Geometry Regressions

The blade geometry for an Xrotor input file is defined in terms of a radial chord and pitch distribution. To obtain realistic distributions for a generic propeller in terms of diameter and pitch, data regressions of actual propeller geometries were developed. The data source was the UIUC propeller database, which includes chord and pitch measurements at 18 radial stations for each tested propeller [31]. The UIUC database contains several styles of propellers, each characterized by different geometric distributions. It was necessary to separate propeller data by style and generate a regression for each one in turn. With few exceptions, the response surfaces achieved strong fits for both the chord and pitch distributions within each style, as is reported by the coefficient of determination (R^2) values (see Table 14). However, it must be noted that some styles only had a few actual propeller samples to fit.

In brief, the regressions define propeller blade geometries that closely match actual commercially available propellers in the diameter range of 2 to 20 inches and a pitch-to-diameter ratio range of 0.35 to 1. The baseline chord distribution obtained from the fit can be scaled uniformly by a solidity scaling factor input.

Table 14: Coefficients of determination for blade geometry response surfaces by style

Style	c/R R^2	Beta R^2	# Data Points	# Propellers
apce	0.995122	0.970313	216	12
apcsf	0.99657	0.975642	126	7
apcsp	0.990685	0.969501	216	12
da4002	0.647233	0.960171	144	8
grcp	0.981982	0.984889	126	7
grcsp	0.97897	0.984527	72	4
grsn	0.996336	0.998469	108	6
gwsdd	0.911552	0.960829	270	15
gwssf	0.991959	0.938464	108	6
kyosho	0.993292	0.989664	90	5
ma	0.986045	0.982649	108	6
mae	0.990317	0.982304	54	3
mas	0.997675	0.989412	126	7
mi	0.967174	0.947476	54	3

A.3.2 Blade Aerodynamic Parameter Estimation

The second set of information for an Xrotor input file is the blade airfoil aerodynamics. Xrotor requires values for the minimum and maximum c_l, critical Mach number, c_m, and model parameters defining a lift curve and a drag polar. The lift model gives c_l as a linear function of angle of attack α (see Eqn. (52)).

$$c_l(\alpha, M) = \frac{CL0 + dCLdA * \alpha}{\sqrt{1 - M^2}} \tag{52}$$

$CL0$ is the value of c_l at $\alpha = 0$, and $dCLdA$ is the slope of the lift curve in the linear region.

Eqn. (53) repeats Xrotor's model of airfoil c_d as a quadratic function of c_l.

$$c_d(c_l, Re) = (CD0 + CD2 * (c_l - CLCD0)^2) * \left(\frac{Re}{REref}\right)^{REexp} \tag{53}$$

$CD0$ is the minimum value of c_d, $CLCD0$ is the value of c_l where $c_d = CD0$, and $CD2$ is the derivative of c_d with respect to c_l^2. $REref$ is the reference Reynold's number at which the previous parameters were notionally measured, while $REexp$ is a power law coefficient modeling the variation in the parameters due to changing Reynolds number.

The aerodynamic parameters were estimated from an airfoil analysis conducted in the Drela-Youngren program, Xfoil [34]. In order to hasten development, only one airfoil was studied. The Clark Y airfoil was selected for this purpose, as it is the airfoil used on most APC style propellers. Xfoil drag polar results were obtained for the Clark Y at Reynolds numbers between 55,000 and 300,000. Aerodynamic parameters were then extracted visually and by fitting the data to Xrotor's lift and drag models. These estimates are reported in Table 15. The lift curve parameters were selected as the average slope and intercept of linear fits applied to each data series within the angle of attack interval of [-2^{o}, 8^{o}] (see Figure 74). Drag polar parameters were determined by fitting the data to the model function using a nonlinear regression solver. Unfortunately, the drag polar data did not readily conform to the model, as depicted in Figure 75. This was particularly the case at lower Reynold's numbers. The fit captures merely the rough order of magnitude of drag, and the general trend.

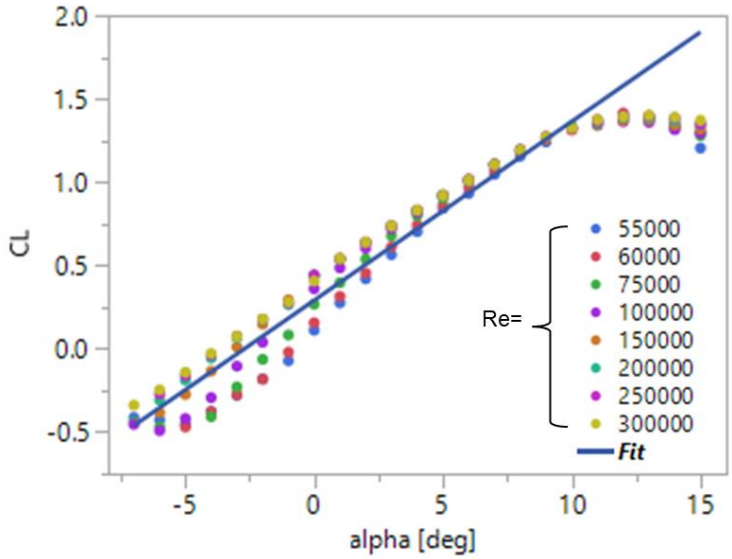

Figure 74: Clark Y lift curve data and fit

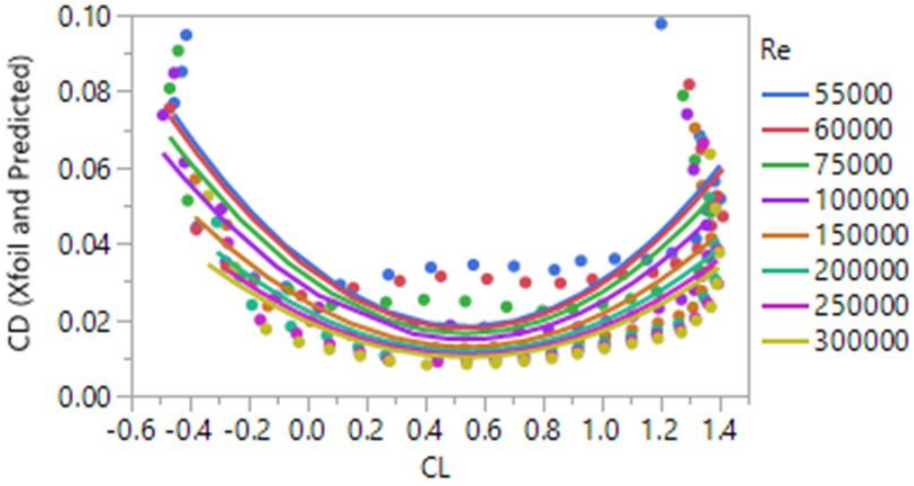

Figure 75: Clark Y drag data and fit

Table 15: Xrotor aerodynamic parameter estimates for Clark Y airfoil

Parameter	Description	Value	Notes
$CL0$	c_l at AoA=0	0.2907372	Average value
$dCLdA$	Lift curve slope	0.1073 1/º	Average value
$CLmin$	Min c_l	-0.4	From visual inspection
$CLmax$	Max c_l	1.4	From visual inspection
$CD0$	Min c_d	0.0101	Fit from nonlinear regression
$CD2$	Induced drag coeff.	0.03164	Fit from nonlinear regression
$CLCD0$	c_l for min c_d	0.5398	Fit from nonlinear regression
$REref$	Reference Re	300000	Fit from nonlinear regression
$REexp$	Drag-Re power law factor	-0.34287	Fit from nonlinear regression
$dCLdAstall$	Unclear from Xrotor documentation	0.1	From other sample Xrotor input files
$dCLstall$	Unclear from Xrotor documentation	0.1	From other sample Xrotor input files
$CMconst$	Moment coefficient	-0.06	From visual inspection
$Mcrit$	Critical Mach number	0.8	assumed

In brief, Xfoil and regression analysis were employed to estimate the aerodynamic parameters of a representative propeller blade airfoil, the Clark Y. The process for obtaining parameter estimates was manual, and parameters were included within the model as hardcoded numbers. Automating this process could increase the power and utility of the tool. For instance, automation would permit a user to select any desired blade airfoil. It might also allow the model to adaptively update aerodynamic estimates according to operating conditions and thereby increase fidelity. This would, however, increase computational time and expense.

A.3.3 Propeller Mass Regression

The propeller input parameters can be used to provide an estimate of the propeller mass. The model incorporates regression of mass as a function of diameter and material presented by Bershadsky [38]. The material options are carbon fiber, nylon, generic plastic,

and wood. The regression was developed for propellers with diameters between 5” and 30”.

A.3.4 Parametric Propeller Model Summary

The implemented parametric propeller model is functional and produces realistic propeller input files for use with Xrotor at minimal computational expense. In so doing, it enables parameterization of the propeller design in terms of continuous variables of pitch and diameter. This constitutes an important modeling capability within the MDA framework, and a valuable contribution to propeller modeling within initial design studies.

A.4 Battery Discharge Model Details

This section details the battery discharge model equations used within the propulsion module. The equations were adapted from [53].

At the highest level, the battery cell voltage is equal to a baseline voltage E_q minus the voltage drop due to internal resistance R_c of the cell. This physics-based equation is shown in Eqn. (54).

$$V_c = E_q - R_c * I_c \tag{54}$$

The baseline voltage varies with the capacity used q (i.e. state of charge). This variation is modeled with Eqn. (55).

$$E_q = E_0 - K * \frac{Q}{Q - q} + A * e^{-B*q} - C * q \tag{55}$$

The coefficients K and Q themselves vary with cell temperature (Eqn. (56) and Eqn. (57)). To simplify analysis, it is assumed that cell temperature is equal to ambient temperature T_a. This assumption neglects the effects of battery heating. Modeling of battery heating lies outside the scope of the present research effort.

$$K = K_0 * e^{\alpha\left(\frac{1}{T_a} - \frac{1}{T_0}\right)} \tag{56}$$

$$Q = Q_p + \gamma(T_a - T_0) \tag{57}$$

The parameter γ models a linear change in effective available capacity due to temperature. Experimental battery discharge data indicate that the relation does not remain linear over large temperature changes. As a result, the assumption of a linear capacity change has a limited range of validity.

The Peukert effect is included within this battery model by adjusting the rated capacity through the relation given in Eqn. (58). p is the Peukert constant and H is the rated discharge time, typically equal to one hour.

$$Q_p = \frac{(Q_0)^p}{(I_c H)^{p-1}} \tag{58}$$

Finally, the cell resistance is itself temperature dependent, modeled by Eqn. (59). Experimental data suggests that internal resistance varies with state-of-charge [52]. This effect is neglected in the implemented model.

$$R = R_0 * e^{\beta\left(\frac{1}{T_a} - \frac{1}{T_0}\right)} \tag{59}$$

If the reference battery internal resistance is unknown, an estimate may be computed. This estimate relies on the assumption that at maximum continuous current draw, the cell voltage will sag from nominal voltage E_0 to cutoff voltage V_{co}. One would expect this to be a conservative assumption, given that at peak current draw, the battery should not get damaged by dropping immediately below the cutoff voltage. The maximum rated current flow is estimated as the product of C-rating and rated capacity. The implied units of C-rating are h^{-1}. The equation for the resistance estimate is shown in Eqn. (60).

$$R_0 = \frac{E_0 - V_{co}}{Q_0 * C_{rating}} \tag{60}$$

Thus, the battery model may be defined in terms of the model parameters E_0, V_{co}, Q_0, C_{rating}, T_0, β, α, p, K_0, γ, A, B, C. Tremblay and Dessaint provide some values of E_0, V_{co}, K_0, A and B for several battery chemistries, which may be used as defaults. Q_0 and C_{rating} are the parameters most easily used for scalable model input, as these are commonly quoted on battery cell datasheets. The remaining parameters $T_0, \alpha, p, \beta, \gamma$, and

C may be given an approximate default value by examining experimental data for cells of a certain chemistry.

The battery discharge equations in their default form may offer generally accurate predictions of the magnitude and behavioral trends of cell voltage as a function of capacity usage, current draw, and temperature. Cell aging effects are not considered, as these are dependent on hysteretic effects, i.e. they will vary depending on the operating conditions at which the cell is charged and discharged [85]. Aging effects can still be simulated by reducing the rated capacity and the C-rating of the cells.

A.5 Conditions for Longitudinal Static Stability of All-Wing Vehicles

This section provides a mathematical derivation of the conditions for longitudinally stable flight of an all-wing vehicle. The derivation closely follows one documented by Mavris and Chakraborty in [19]

Consider the all-wing vehicle in steady level flight depicted in Figure 76. Weight (W) acts downward at the center of gravity. Assume that thrust (T) acts at the center of gravity as well. The pressure and shear distributions on the wing result in a force and a moment. The force is decomposed into lift (L) and drag (D). As the vehicle is not accelerating in any direction, drag equals thrust and lift equals weight. There is a point on the vehicle where the moment (M_{np}) does not vary strongly with angle of attack (α). This is the neutral point, where lift and drag shall be assumed to act, and which has the convenient property that $\frac{\partial M_{np}}{\partial \alpha} = 0$. The center of gravity is located a distance l ahead of the neutral point.

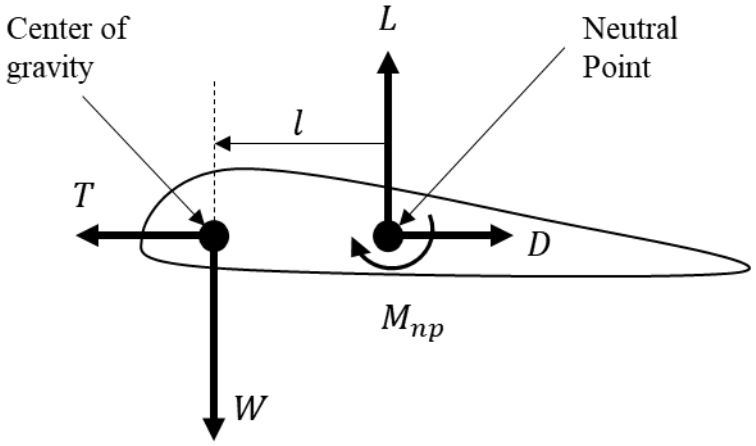

Figure 76: Simplified free-body-diagram of a flying wing

If the aircraft is trimmed, then the sum of moments must equal 0 about any point. Assuming the moment arm of drag is negligible, the sum of moments about the center of gravity is given by Eqn. (61).

$$\sum M = 0 = M_{np} - l * L \tag{61}$$

It is more convenient to continue working with nondimensional coefficients. Dividing both sides of the equation by qSc (dynamic pressure times reference area times reference chord) yields Eqn. (62).

$$C_m = 0 = C_{m_{np}} - \frac{l}{c} C_L \tag{62}$$

Rearranging for l gives:

$$l = \frac{cC_{m_{np}}}{C_L} \tag{63}$$

Eqn. (62) expresses the constraint imposed by static equilibrium. An additional equation comes from the requirement of static stability, which is ensured if $\frac{\partial C_M}{\partial \alpha} = C_{M_\alpha} < 0$. Differentiating Eqn. (62) with respect to angle of attack yields a requirement for static stability:

$$C_{m_\alpha} = \frac{\partial C_{m_{np}}}{\partial \alpha} - \frac{l}{c} \frac{\partial C_L}{\partial \alpha} < 0 \tag{64}$$

It was noted earlier that the neutral point had the convenient property that $\frac{\partial M_{np}}{\partial \alpha} = \frac{\partial C_{m_{np}}}{\partial \alpha} = 0$. Substituting this and Eqn. (63) into Eqn. (64) leads to Eqn. (65)

$$-\frac{C_{m_{np}}}{C_L} \frac{\partial C_L}{\partial \alpha} < 0 \tag{65}$$

For any reasonable airfoil, lift coefficient increases roughly linearly with angle of attack. Thus, $\frac{\partial C_L}{\partial \alpha} > 0$. Since the aircraft is flying steady and level, lift must be counteracting weight, so $C_L > 0$. Thus, one concludes that to achieve both static equilibrium and static stability on a flying wing, it is necessary that:

$$C_{m_{np}} > 0 \tag{66}$$

If this is the case, then it follows from Eqn. (63) that:

$$l > 0 \tag{67}$$

In other words, the moment coefficient about the neutral point must be positive and the center of gravity must be located ahead of the neutral point.

A.6 Performance Modeling Functions

This section documents the equations and assumptions used within each modeling function that was implemented within the performance module.

Steady Level Flight

The steady level flight analysis assumes the aircraft is flying at a constant altitude and speed. Lift and drag are obtained from an aerodynamic analysis that assumes lift equals weight, as indicated in Eqn. (68). Thrust must then equal drag, as Eqn. (69) states. These equations are derived from a force balance based on Figure 77. The current draw is determined by conducting the thrust-prescribed analysis within the propulsion module. The integration can be terminated in three manners. First, it may end after the aircraft has flown a prescribed distance. Second, it may end if it has flown for a prescribed time. Finally, it may end if the cell voltage sags to a fraction of the cutoff voltage.

$$m_{veh} a_v = L - W = 0 \tag{68}$$

$$m_{veh} a_h = T - D = 0 \tag{69}$$

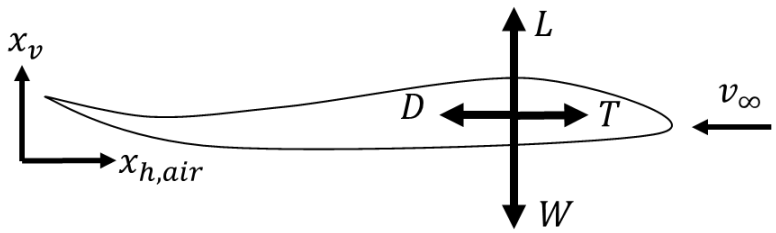

Figure 77: Free body diagram for steady level flight

Steady Level Turn

In steady level turn, the aircraft executes a banked turn at a constant speed and altitude. Eqn. (69) remains valid for this segment. The turn must be defined in terms of two of the following four parameters: turn rate ω, turn radius R, airspeed v_∞, and load factor n. Given two of the parameters, the other two may be calculated using rearranged forms of Eqn. (70)

and Eqn. (71), which are derived from the horizontal force balance in Eqn. (73). g is gravitational acceleration. Load factor and bank angle ϕ are related through Eqn. (72). Lift is computed from Eqn. (74), which is derived from the free body diagram depicted in Figure 78. The method terminates once the turn, defined by an input angle of turn ζ, is completed. The total air distance covered will then equal $R\zeta$. Ground track distance is computed by evaluating Eqn. (75). This equation is an application of the general expression for the length of a parameterized curve. It assumes that at the start of the turn, the aircraft is facing directly into the wind.

$$v_\infty = \frac{g\sqrt{n^2-1}}{\omega} \tag{70}$$

$$\omega = \frac{v_\infty}{R} \tag{71}$$

$$n = \frac{1}{\cos(\phi)} \tag{72}$$

$$L\sin(\phi) = m_{veh}\omega^2 R \tag{73}$$

$$L\cos(\phi) = W \tag{74}$$

$$x_g = \int_0^{\theta_f} \sqrt{(Rcos(\theta) + v_{wind})^2 + \left(Rsin(\theta)\right)^2}\, d\theta \tag{75}$$

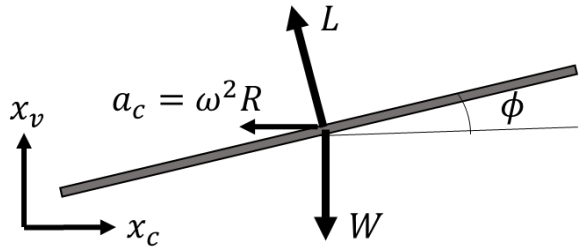

Figure 78: Free body diagram for steady level turn

Accelerated Level Flight

A level acceleration analysis is included, which assumes the aircraft starts from an initial speed and accelerates to a final speed without changing altitude. The analysis approach assumes that at each integration step, the vehicle is in a state of quasi-equilibrium. Eqn. (68) still applies in this case. Available thrust is determined by a throttle-prescribed analysis at a constant throttle setting. Horizontal acceleration is found from Eqn. (69), ignoring the constraint that thrust and drag are equal. The method terminates when either the acceleration is zero (i.e. the aircraft has reached maximum speed) or the vehicle has attained the prescribed final speed.

Maximum Speed

The performance module can solve for a maximum vehicle speed at a given altitude and throttle setting. It does so by varying airspeed and computing both available and required thrust. The speed at which required thrust equals available thrust is the maximum speed. A numerical root finding method is employed to converge on this value. The computation can take up to several seconds to complete if a full Xrotor analysis is employed to predict available thrust. A related computation method can generate a plot of required and available thrust as a function of airspeed. An example of such a plot is depicted in Figure 79.

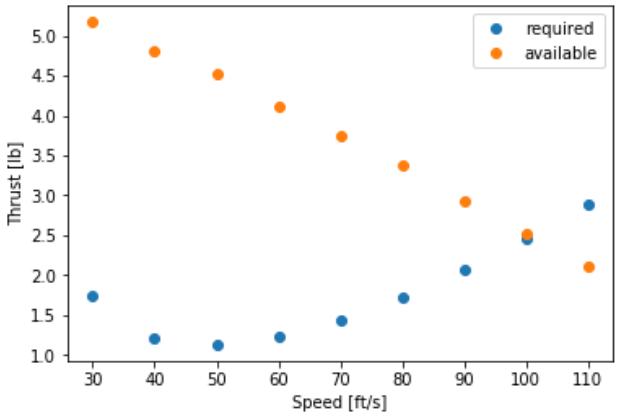

Figure 79: Thrust required and available vs. speed at sea level for a generic flying

Steady Climb

The steady climb analysis assumes the vehicle climbs at a steady incoming airspeed and a constant throttle setting from an initial altitude to a final altitude. The equations of motion are shown in Eqn. (76) and Eqn. (77). They assume thrust and drag are aligned, as shown in Figure 80. The climb angle γ is determined using a numerical root finding method. This computation may result in three outcomes. First, a positive value of γ satisfying both equations may be found. Second, a negative value of γ may be found, indicating that at the given throttle setting, the vehicle is incapable of producing enough thrust to climb. Finally, the vehicle may be producing so much thrust that the aircraft would accelerate vertically. Both latter outcomes result in error messages.

$$m_{veh}a_h = (T - D)\cos(\gamma) - L\sin(\gamma) = 0 \tag{76}$$

$$m_{veh}a_v = (T - D)\sin(\gamma) + L\cos(\gamma) - W = 0 \tag{77}$$

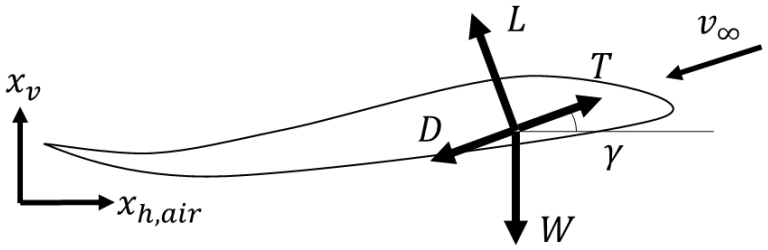

Figure 80: Free body diagram for steady climb

Maximum Rate of Climb

Maximum rate of climb is computed using the same equations used for the steady climb analysis. To simplify computations, the climb speed is set to equal the speed for steady level flight at which $(C_L)^{1.5}/C_D$ is maximized. This speed minimizes power consumption, thereby maximizing excess power, and hence rate of climb. The values of C_L and C_D for maximum $(C_L)^{1.5}/C_D$ are computed using the stick-fixed drag polar, rather than a trim AVL analysis. A related method outputs the maximum rate of climb for a range of altitudes at a fixed throttle setting. This table can be used to estimate the vehicle's service ceiling. An example is included in Figure 81.

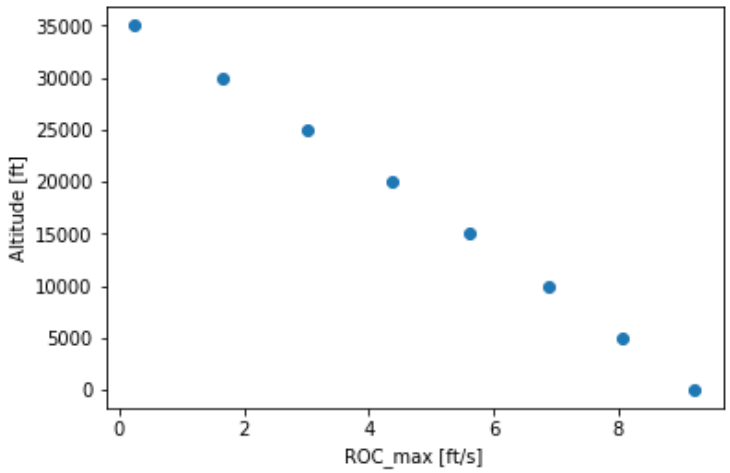

Figure 81: Altitude vs. maximum rate of climb for a generic flying wing

Steady Descent – Unpowered

The unpowered steady descent model assumes the vehicle is descending at a steady rate of -$v_\infty \sin(\gamma)$. The force balance equations are the same as those for steady climb, except that $T = 0$. The climb angle is computed through a convergence loop. As the vehicle is unpowered, the current draw is simply equal to the current required to power any non-propulsive subsystems. The analysis ends when the vehicle descends to a target altitude.

Launch

A simple launch analysis is included for vehicles that are to be either catapult or hand launched. Initial velocity is modeled using the assumption that a potential launch energy e_{launch} is completely converted into kinetic energy of the vehicle. The initial speed of the vehicle v_{launch} with respect to the ground is calculated using Eqn. (78)

$$v_{launch} = \sqrt{\frac{2e_{launch}}{m_{veh}}} \tag{78}$$

The vehicle may be launched at an initial angle γ_{launch}. It is assumed that the vehicle is operating at $C_{L_{max}}$ and a constant throttle setting for the duration of the analysis. Vertical and horizontal accelerations are computed from Eqn. (79) and Eqn. (80), which are dynamic versions of the force balance equations for steady climb. The method terminates when the vertical acceleration of the aircraft is greater than zero and the vehicle is at or above its starting altitude. If the vehicle drops by more than an input distance below its starting point, the analysis exits with an error.

$$m_{veh} a_h = (T - D)\cos(\gamma) - L\,sin(\gamma) \tag{79}$$

$$m_{veh} a_v = L\cos(\gamma) - W + (T - D)\sin(\gamma) \tag{80}$$

Takeoff

A takeoff analysis method is included which codifies the equations and rules of thumb presented by Raymer [9]. Ground roll, rotation, transition, and obstacle clearance are all included within the analysis. Drag is predicted using the stick-fixed drag polar rather than an AVL analysis. Changes to the quadratic drag polar coefficient due to ground effect are considered, but using a regression presented by Raymer, which may only be valid for large conventional aircraft. An input value of wing incidence angle is used within an AVL analysis to estimate the vehicle's C_L during ground roll. The analysis exits when the vehicle clears the obstacle.

Landing Ground Roll – Unpowered

The landing ground roll analysis presented by Raymer is also included within the module. It assumes that the aircraft is unpowered, i.e. providing no thrust. Both free roll and braking roll are included. As in the takeoff analysis, drag is computed from the stick-fixed drag polar, and the C_L during ground roll is estimated to be the corresponding value at an input incidence angle. The method ends when the vehicle's ground speed is zero.

A.7 Range and Endurance for Electric Aircraft

This section derives the Breguet range and endurance equations for electric vehicles. It also includes derivations of battery weight fractions for maximum range and endurance.

Breguet Endurance Equation for Electric Aircraft

The core assumption for deriving simplified range and endurance equations is that the aircraft is flying steady and level. Lift L equals weight W, and thrust T equals drag D. Moreover, the weight of the aircraft does not change over time since it is electrically powered:

$$L = W \tag{81}$$

$$T = D \tag{82}$$

The rate at which the aircraft consumes onboard electrical energy E is equal to the thrust power divided by a propulsive efficiency factor η (see Eqn. (83)). As an additional simplification, this efficiency is considered constant in time. v_∞ is the vehicle's airspeed.

$$\frac{dE}{dt} = \frac{Tv_\infty}{\eta} = \frac{Dv_\infty}{\eta} \tag{83}$$

Drag is related to the drag coefficient through Eqn. (84). ρ is the air density and S is the invariant reference area. It is assumed that density does not change over the course of a steady level flight segment.

$$D = \frac{1}{2}\rho v_\infty^2 C_D S \tag{84}$$

Similarly, the airspeed is related to the lift coefficient through Eqn. (85).

$$v_\infty = \sqrt{\frac{W}{S}\frac{2}{\rho C_L}} \tag{85}$$

Substituting the previous two relations into Eqn. (83) yields Eqn. (86):

$$\frac{dE}{dt} = \sqrt{\frac{2}{\rho S}} \frac{1}{\eta} \frac{C_D}{C_L^{1.5}} W^{1.5} \tag{86}$$

Eqn. (86) can then be rearranged for the time differential dt:

$$dt = \eta \sqrt{\frac{\rho S}{2}} \frac{C_L^{1.5}}{C_D} W^{-1.5} dE \tag{87}$$

The endurance expression in Eqn. (88) is obtained by integrating Eqn. (87). Δt is the endurance, and ΔE is the total energy consumed.

$$\Delta t = \eta \sqrt{\frac{\rho S}{2}} \frac{C_L^{1.5}}{C_D} W^{-1.5}\, \Delta E \tag{88}$$

To a gross degree of accuracy, the total energy available in a battery is proportional to its weight W_{batt} through a value of mass specific energy e_{batt}. Thus, ΔE can be expressed with Eqn. (89), where g is gravitational acceleration and K is a factor between 0 and 1 determining what fraction of available energy was used in the loiter.

$$\Delta E = \frac{K\, W_{batt}\, e_{batt}}{g} \tag{89}$$

Substituting this expression into Eqn. (88) gives a final form of the Breguet endurance equation for electric aircraft:

$$\boldsymbol{\Delta t = \frac{\eta}{g} \sqrt{\frac{\rho S}{2}} \frac{C_L^{1.5}}{C_D} K e_{batt} \frac{W_{batt}}{W^{1.5}}} \tag{90}$$

Breguet Range Equation for Electric Aircraft

Alternatively, Eqn. (87) can be adjusted to provide an expression for a differential distance ds:

$$ds = v_\infty \, dt = \frac{\eta}{W} \frac{C_L}{C_D} dE \tag{91}$$

Integrating Eqn. (91) and substituting in Eqn. (89) as before results in the Breguet range equation for electric aircraft:

$$\boldsymbol{\Delta s = \frac{\eta}{g} \frac{W_{batt}}{W} \frac{C_L}{C_D} K \, e_{batt}} \tag{92}$$

Battery Weight Fraction for Maximum Endurance:

From before, total aircraft weight may be decomposed into battery weight W_{batt} and remaining weight W_e:

$$W = W_{batt} + W_e \tag{93}$$

The value of W_{batt} that maximizes endurance is calculated by taking the partial derivative of Eqn. (90) with respect to W_{batt} and setting it equal to zero:

$$\frac{\partial \Delta t}{\partial W_{batt}} = \frac{\eta}{g} \sqrt{\frac{\rho S}{2}} \frac{C_L^{1.5}}{C_D} K e_{batt} \frac{(W_{batt} + W_e)^{1.5} - W_{batt} * 1.5 * (W_{batt} + W_e)^{0.5}}{(W_{batt} + W_e)^{2.25}} = 0$$

$$(W_{batt} + W_e)^{1.5} = W_{batt} * 1.5 * (W_{batt} + W_e)^{0.5}$$

$$W_{batt} + W_e = W_{batt} * 1.5$$

$$W_{batt} = 2W_e$$

This result suggests that for a set value of total aircraft weight, endurance is maximized if two thirds of that weight stores the electrical energy:

$$\boldsymbol{\left(\frac{W_{batt}}{W}\right)_{opt.End} = \frac{2}{3}} \tag{94}$$

Battery Weight Fraction for Maximum Range:

Similarly, the value of W_{batt} that maximizes range can be calculated by taking the partial derivative of Eqn. (92) with respect to W_{batt} and setting it equal to zero:

$$\frac{\partial \Delta s}{\partial W_{batt}} = \frac{\eta}{g}\frac{C_L}{C_D} K\, e_{batt} \frac{W_e}{(W_{batt} + W_e)^2} = 0$$

$$W_e = 0$$

This result suggests that for a set value of total aircraft weight, range is maximized if the entire weight stores the electrical energy:

$$\left(\frac{\boldsymbol{W_{batt}}}{\boldsymbol{W}}\right)_{\boldsymbol{opt.Range}} = \mathbf{1} \tag{95}$$

REFERENCES

[1] J. D. Barton, "Fundamentals of Small Unmanned Aircraft Flight," *Johns Hopkins APL Technical Digest*, vol. 31, no. 2, pp. 132-149, 2012.

[2] J. Anderson, "6.20 Uninhabited Aerial Vehicles (UAVs)," in *Introduction to Flight*, New York, McGraw Hill, 2015.

[3] J. Gundlach, Designing Unmanned Aircraft Systems a Comprehensive Approach, Reston: American Institute of Aeronautics and Astronautics, 2012.

[4] A. Watts, V. Ambrosia and E. Hinkley, "Unmanned Aircraft Systems in Remote Sensing and Scientific Research: Classification and Considerations of Use," *Remote Sensing*, vol. 4, no. 6, pp. 1671-1692, 2012.

[5] Teal Group, "World Civil Unmanned Aerial Systems - 2018 Market Profile and Forecast," Teal Group, Fairfax, 2018.

[6] K. Nickel and M. Wohlfahrt, Tailless Aircraft in Theory and Practice, Washington DC: American Institute of Aeronautics and Astronautics, 1994.

[7] J. Anderson, Aircraft Performance and Design, New Delhi: McGraw-Hill, 2012.

[8] R. Colgren and R. Loschke, "Effective Design of Highly Maneuverable Tailless Aircraft," *Journal of Aircraft*, vol. 45, no. 4, pp. 1441-1449, 2008.

[9] D. Raymer, Aircraft Design - A Conceptual Approach, Washington DC: American Institute of Aeronautics and Astronautics, 1992.

[10] NASA, "NASA Armstrong Fact Sheet: Helios Prototype," NASA, 28 2 2014. [Online]. Available: https://www.nasa.gov/centers/armstrong/news/FactSheets/FS-068-DFRC.html. [Accessed 18 6 2019].

[11] J. Gundlach, "4.4.5 Flying Wings," in *Designing Unmanned Aircraft Systems: A Comprehensive Approach*, Reston, American Institute of Aeronautics and Astronautics, 2012, pp. 120-124.

[12] Northrop Grumman, "The B-2 Spirit Stealth Bomber Turns 25," 9 July 2014. [Online]. Available: https://www.northropgrumman.com/Capabilities/B2SpiritBomber/Documents/pageDocuments/B-2_25th_Anniversary_Fact_Sheet.pdf. [Accessed 5 6 2019].

[13] J. Northrop, "The Developmet of All-Wing Aircraft," *The Journal of the Royal Aeronautical Society,* vol. 51, no. 438, pp. 481-510, 1947.

[14] M. Hepperle, "Electric Flight – Potential and Limitations," German Aerospace Center, Braunschweig, 2013.

[15] R. Rezende and J. Barros, "General Aviation 2025 - A Study in Electric Propulsion," in *AIAA Propulsion and Energy Forum*, 2018.

[16] J. Gundlach, "Propulsion Systems," in *Designing Unmanned Aircraft Systems: A Comprehensive Approach*, Reston, American Institute of Aeronautics and Astronautics, 2012, pp. 289-352.

[17] PH0NO, "Power sources for portable operation," Blogger, 24 5 2016. [Online]. Available: http://www.ph0no.net/2016/05/power-sources-for-portable-operation.html. [Accessed 27 6 2019].

[18] "AIAA Design, Build, Fly," American Institute of Aeronautics and Astronautics, 2019. [Online]. Available: https://www.aiaa.org/dbf. [Accessed 27 6 2019].

[19] D. Mavris and I. Chakraborty, "Aircraft Flight Dynamics, Stability, and Control," Aerospace Systems Design Lab, Georgia Institute of Technology, Atlanta, 2018.

[20] J. Sobieszczanski-Sobieski, "Multidisciplinary aerospace design optimization: survey of recent developments," *Structural Optimization,* vol. 14, no. 1, pp. 1-23, 1997.

[21] J. R. R. A. Martins and A. B. Lambe, "Multidisciplinary Design Optimization: A Survey of Architectures," *AIAA Journal,* vol. 51, no. 9, pp. 2049-2075, 2013.

[22] "ModelCenter Integrate," Phoenix Integration, 2019. [Online]. Available: https://www.phoenix-int.com/product/modelcenter-integrate/. [Accessed 18 9 2019].

[23] "Matlab," MathWorks, 2019. [Online]. Available: https://www.mathworks.com/products/matlab.html. [Accessed 18 9 2019].

[24] "Python," Python Software Foundation, 2019. [Online]. Available: https://www.python.org/. [Accessed 28 6 2019].

[25] J. Gray, J. Hwang, J. Martins, K. Moore and B. Naylor, "OpenMDAO: an open-source framework for multidisciplinary design, analysis, and optimization," *Structural and Multidisciplinary Optimization,* vol. 59, no. 4, pp. 1075-1104, 2019.

[26] "OpenVSP: Vehicle Sketch pad - NASA open source parametric geometry," OpenVSP, [Online]. Available: http://openvsp.org/. [Accessed 27 6 2019].

[27] L. Traub, "Range and Endurance Estimates for Batter-Powered Aircraft," *Journal of Aircraft,* vol. 48, no. 2, pp. 703-707, 2011.

[28] "MotoCalc," Capable Computing Inc., 2018. [Online]. Available: http://www.motocalc.com/. [Accessed 26 6 2019].

[29] "eCalc," Solution for All Markus Mueller, [Online]. Available: https://www.ecalc.ch/. [Accessed 26 6 2019].

[30] C. Johnson, Interviewee, *Propulsion Modeling Discussion.* [Interview]. 14 3 2019.

[31] J. Brandt, R. Deters, G. Ananda and M. Selig, "UIUC Propeller Data Site," UIUC Applied Aerodynamics Group, [Online]. Available: https://m-selig.ae.illinois.edu/props/propDB.html. [Accessed 27 6 2019].

[32] "Performance Data," Advanced Precision Composites, [Online]. Available: https://www.apcprop.com/technical-information/performance-data/. [Accessed 27 6 2019].

[33] M. Drela and H. Youngren, "Qprop," Massachusetts Institute of Technology, [Online]. Available: http://web.mit.edu/drela/Public/web/qprop/. [Accessed 27 6 2019].

[34] M. Drela and H. Youngren, "Xfoil," Massachusetts Institute of Technology, [Online]. Available: https://web.mit.edu/drela/Public/web/xfoil/. [Accessed 27 6 2019].

[35] M. Drela and H. Youngren, "Xrotor Download Page," Massachusetts Institute of Technology, [Online]. Available: http://web.mit.edu/drela/Public/web/xrotor/. [Accessed 27 6 2019].

[36] P. Carter, "Crotor: Xrotor on Steroids," Philip Carter, ESOTEC Developments, [Online]. Available: http://www.esotec.org/sw/crotor.html. [Accessed 27 6 2019].

[37] M. Drela and H. Youngren, "DFDC: Ducted Fan Design Code," Massachusetts Institute of Technology, [Online]. Available: http://web.mit.edu/drela/Public/web/dfdc/. [Accessed 27 6 2019].

[38] D. Bershadsky, "Electric Multirotor Design and Optimization," Georgia Institute of Technology, Atlanta, 2017.

[39] M. Drela, "Second-Order DC Electric Motor Model," Massachusetts Institute of Technology, Cambridge, 2006.

[40] "Innov8tive Designs," Innov8tive Designs, 2019. [Online]. Available: http://innov8tivedesigns.com/parts/brushless-motors?manufacturer=10. [Accessed 26 6 2019].

[41] J. Winslow, V. Hrishikeshavan and I. Chopra, "Design Methodology for Small-Scale Unmanned Quadrotors," *Journal of Aircraft*, vol. 55, no. 3, pp. 1062-1070, 2018.

[42] A. Harrington and C. Kroninger, "Characterization of Small DC Brushed," Army Research Laboratory, Aberdeen MD, 2013.

[43] Dejan, "How Brushless Motor and ESC Work," How To Mechatronics, [Online]. Available: https://howtomechatronics.com/how-it-works/how-brushless-motor-and-esc-work/. [Accessed 26 6 2019].

[44] C. Green, "Modeling and Test of the Efficiency of Electronic Speed Controllers for Brushless DC Motors," California Polytechniw State University, San Luis Obispo, 2015.

[45] A. Gong, R. Macneill and D. Verstraete, "Performance Testing and Modeling of a Brushless DC Motor, Electronic Speed Controller and Propeller for a Small UAV Application," in *2018 Joint Propulsion Conference*, 2018.

[46] G. Elert, "Electric Resistance," The Physics Hypertextbook, [Online]. Available: https://physics.info/electric-resistance/. [Accessed 27 6 2019].

[47] O. Liang, "FPV Wires and Connector - Sizes, Current, and Selection," Oscar Liang, May 2014. [Online]. Available: https://oscarliang.com/wire-awg-chart-quadcopter-rc/. [Accessed 30 9 2019].

[48] "Current Carrying Capcities for FEP (Teflon) and Silicone Inuslated Wire," Wiremax, a Heico Company, [Online]. Available:

http://catalog.connectronicscorp.com/Asset/WIREMAX-conductor-CURRENT--2-.pdf. [Accessed 30 9 2019].

[49] "BU-402: What Is C-rate?," Battery University, 9 3 2017. [Online]. Available: https://batteryuniversity.com/learn/article/what_is_the_c_rate. [Accessed 18 10 2019].

[50] MIT Electric Vehicle Team, "A Guide to Understanding Battery Specifications," MIT, Cambridge, 2008.

[51] O. Tremblay and L.-A. Dessaint, "Experimental Validation of a Battery Dynamic Model for EV," *World Electric Vehicle* , vol. 3, pp. 289-298, 2009.

[52] D. Wang, Y. Bao and J. Shi, "Online Lithium-Ion Battery Internal Resistance Measurement Application in State-of-Charge Estimation Using the Extended Kalman Filter," *Energies*, vol. 10, no. 9, p. 1284, 2017.

[53] "Battery: Generic Battery Model," MathWorks, [Online]. Available: https://www.mathworks.com/help/physmod/sps/powersys/ref/battery.html. [Accessed 19 9 2019].

[54] M. McMahon, "pywinauto," Mark McMahon and Contributors, 2018. [Online]. Available: https://pywinauto.readthedocs.io/en/latest/index.html. [Accessed 25 9 2019].

[55] M. Hepperle, "Airfoils for Flying Wings," Martin Hepperle, 21 5 2018. [Online]. Available: https://www.mh-aerotools.de/airfoils/. [Accessed 4 10 2019].

[56] S. Hartmut, "Airfoil Database - Tailless and Flying Wings," Aerodesign, 3 9 2001. [Online]. Available: https://www.aerodesign.de/english/profile/profile_s.htm#mh45. [Accessed 4 10 2019].

[57] H. Quabeck, "Profilkoordinated und Polaren," HQ-Modellflug, [Online]. Available: http://www.hq-modellflug.de/koordinatenframe.htm. [Accessed 4 10 2019].

[58] R. Horten and W. Horten, "Ten Years Development of the Flying Wing High-Speed Fighter," Chance Vought Aircraft, Stratford, 1946.

[59] M. Drela and H. Youngren, "AVL," Massachusetts Institute of Technology, [Online]. Available: http://web.mit.edu/drela/Public/web/avl/. [Accessed 27 6 2019].

[60] "UIUC Airfoil Data Site," UIUC Applied Aerodynamics Group, [Online]. Available: https://m-selig.ae.illinois.edu/ads.html. [Accessed 1 10 2019].

[61] "17.1 suprocess - Suprocess management," Python, [Online]. Available: https://docs.python.org/2/library/subprocess.html. [Accessed 1 10 2019].

[62] C. Johnson, "Appendix B: AVL Tutorial," Georgia Institute of Technology, Atlanta, 2015.

[63] "Fluid - Lecture 5 Notes," Massachusetts Institute of Technology, [Online]. Available: http://web.mit.edu/16.unified/www/SPRING/fluids/Spring2008/LectureNotes/f05.pdf. [Accessed 1 10 2019].

[64] "scikit-aero 0.1," Pypi, 25 11 2012. [Online]. Available: https://pypi.org/project/scikit-aero/. [Accessed 7 10 2019].

[65] T. Benson, "Air Viscosity - Sutherland's Formula," NASA, [Online]. Available: https://www.grc.nasa.gov/www/BGH/viscosity.html. [Accessed 7 10 2019].

[66] H. Karakas, E. Koyuncu and G. Inalhan, "ITU Tailless UAV Design," *Journal of Intelligent & Robotic Systems,* vol. 69, no. 1, pp. 131-146, 2013.

[67] G. Steinfelt and U. Ringertz, "Yaw Control of a Tailless Aircraft Configuration," *Journal of Aircraft,* vol. 47, no. 5, pp. 1807-1811, 2010.

[68] C. John, "Jack Northrop and the Flying Wing," *Air Force Magazine,* pp. 68-73, February 2017.

[69] "Horten Ho 229," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Horten_Ho_229. [Accessed 21 6 2019].

[70] "Northrop YB-49," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Northrop_YB-49. [Accessed 21 6 2019].

[71] Northrop Grumman, "B-2 Spirit Bomber Gallery," Northrop Grumman Corporation, [Online]. Available: https://www.northropgrumman.com/MediaResources/Pages/Photo.aspx?pid%3DBS-10001_019%26rel%3D%2F%26name%3DPhotos. [Accessed 21 6 2019].

[72] "X-47B UCAS “Flipbook” Gallery," Northrop Grumman Corporation, [Online]. Available:

https://www.northropgrumman.com/Capabilities/X47BUCAS/Pages/Flipbook.aspx. [Accessed 21 6 2019].

[73] "Boeing X-45," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Boeing_X-45. [Accessed 21 6 2019].

[74] C. Pocock, "BAE Systems Awaits UK Combat Air Strategy," The Convention News Company, 12 7 2018. [Online]. Available: https://www.ainonline.com/aviation-news/aerospace/2018-07-12/bae-systems-awaits-uk-combat-air-strategy. [Accessed 21 6 2019].

[75] G. Whitwell, "Cutting and Packing," University of Nottingham, Nottingham, 2004.

[76] H. Gehring, K. Menschner and M. Meyer, "A computer-based heuristic for packing pooled shipment containers," *European Journal of Operational Research,* vol. 44, no. 2, pp. 277-288, 1990.

[77] P. Gilmore and R. Gomory, "Multistage cutting stock problems of two and more dimensions," *Operations Research,* vol. 13, no. 1, pp. 94-120, 1965.

[78] R. Haessler and B. Talbot, "Load planning for shipments of low density products," *European Journal of Operational Research,* vol. 44, no. 2, pp. 289-299, 1990.

[79] E. Hopper and B. Turton, "An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem," *European Journal of Operational Research,* vol. 128, no. 1, pp. 34-57, 2001.

[80] A. Crispin, P. Clay, G. Taylor, T. Bayes and D. Reedman, "Genetic Algorithm Coding Methods for Leather Nesting," *Applied Intelligence,* vol. 23, no. 1, pp. 9-20, 2005.

[81] S. Anand, C. McCord, R. Sharma and T. Balachander, "An integrated machine vision based system for solving the nonconvex cutting stock problem using genetic algorithms," *Journal of Manufacturing Systems,* vol. 18, no. 6, pp. 296-415, 1999.

[82] G. Fadel, M. Wiecek, G. Fasano and J. Pinter, "Packing Optimization of Free-Form Objects in Engineering Design," in *Optimized Packings with Applications*, Cham, Springer International Publishing, 2015, pp. 37-66.

[83] P. Grignon, J. Wodziak and G. Fadel, "Bi-objective optimization of components packing using a genetic algorithm," in *6th Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, 1996.

[84] "About Us," TMachines, 2017. [Online]. Available: https://www.tmachines.com/about-us/. [Accessed 18 6 2019].

[85] P. Rong and M. Pedram, "An Analytical Model for Predicting the Remaining Battery Capacity of Lithium-Ion Batteries," *IEEE Transactions on VLSI Systems,* vol. 14, no. 5, pp. 441-451, 2006.

[86] G. Elert, "Electric Resistance," The Physics Hypertextbook, [Online]. Available: https://physics.info/electric-resistance/. [Accessed 18 10 2019].